

Inna Shingareva  
Carlos Lizárraga-Celaya

# Maple and Mathematica

A Problem Solving Approach  
for Mathematics



SpringerWienNewYork



Inna Shingareva  
Carlos Lizárraga-Celaya

# Maple and Mathematica

A Problem Solving Approach  
for Mathematics

SpringerWienNewYork

**Dr. Inna Shingareva**

Department of Mathematics, University of Sonora, Sonora, Mexico

**Dr. Carlos Lizárraga-Celaya**

Department of Physics, University of Sonora, Sonora, Mexico

This work is subject to copyright.

All rights are reserved, whether the whole or part of the material is concerned, specifically those of translation, reprinting, re-use of illustrations, broadcasting, reproduction by photocopying machines or similar means, and storage in data banks.

**Product Liability:** The publisher can give no guarantee for all the information contained in this book. This does also refer to information about drug dosage and application thereof. In every individual case the respective user must check its accuracy by consulting other pharmaceutical literature. The use of registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

© 2007 Springer-Verlag/Wien

Printed in Austria

SpringerWienNewYork is part of  
Springer Science + Business Media  
[springer.at](http://springer.at)

Typesetting: Camera ready by the authors

Printing: Holzhausen Druck und Medien GmbH, 1140 Vienna, Austria

Printed on acid-free and chlorine-free bleached paper

SPIN: 12078479

Library of Congress Control Number: 2007932621

ISBN 978-3-211-73264-9 SpringerWienNewYork

*To our parents,  
with infinite admiration, love, and gratitude.*

## Preface

It is well known that computer algebra systems have revolutionized teaching and the learning processes in mathematics, science, and engineering, allowing students to computationally investigate complicated problems to find exact or approximate analytic solutions, numeric solutions, and illustrative two- and three-dimensional graphics.

Since the 1960s there has existed individual packages for solving specific analytic, numerical, graphical and other problems. The need to solve all those problems with the aid of a single system, has led to the idea of construction of a modern general purpose computer algebra system. The first two papers describing analytic calculations realized on a computer were published in 1953 [7]. In the early 1970s, *systems of analytic computations (SAC)*, or *computer algebra systems (CAS)*, began to appear.

Computer algebra systems are computational interactive programs that facilitate symbolic mathematics and can handle other type of problems. The first popular systems were Reduce, Derive, and Macsyma, which are still commercially available. Macsyma was one of the first and most mature systems. It was developed at the Massachusetts Institute of Technology (MIT), but practically its evolution has stopped since the summer of 1999. A free software version of Macsyma, Maxima, is actively being maintained.

To the present day, there have been developed more than a hundred computer algebra systems [7], [18]. Among these we can find Axiom, Derive, Maxima, Maple, Mathematica, Matlab, MuMATH, MuPAD, Reduce, etc. All these systems can be subdivided into specialized and general-purpose computer algebra systems ([7], [18], [2]).

In this book we consider the two general purpose computer algebra systems, *Maple* and *Mathematica* which appeared around 1988; today both being the most popular, powerful, and reliable systems that are

used worldwide by students, mathematicians, scientists, and engineers. It is not, however, our intention to prove or assert that either *Maple* or *Mathematica*, is better than the other. Following our experience in working with these two computer algebra systems, we have structured this book in presenting and work in parallel in both systems. We note that *Mathematica*'s users can learn *Maple* quickly by finding the *Maple* equivalent to *Mathematica* functions, and vice versa. Additionally, in many research problems it is often required to make independent work and compare the results obtained using the two computer algebra systems, *Maple* and *Mathematica*.

Let us briefly describe these two computer algebra systems.

The computer algebra system *Maple* (see [8]) was developed initially at the University of Waterloo in the 1980s as a research project. At the beginning, the software development group discussed the idea of how to develop a system like Macsyma. Now *Maple* has a rich set of functions that makes it comparable to Macsyma in symbolic power [18] and has a leading position in education, research, and industry. It incorporates the best features of the other systems, it is written in C language and has been ported to the major operating systems.

The *Maple* system evolution has been more community oriented, that is, it allows community participation in the development and incorporation of new tools. Today, modern *Maple* offers three main systems: *The Maple Application Center*, *Maple T.A.*, and *MapleNet*. *The Maple Application Center* offers free resources, for example, research problems, simple classroom notes, entire courses in calculus, differential equations, classical mechanics, linear algebra, etc. *Maple T.A.*, provided for Web-based testing and assessment, includes access control, interactive grade book, algorithmic question generation, and intelligent assessment of mathematical responses. *MapleNet* is a full environment for online e-learning, providing an interactive Web-based environment for education.

In contrast, *Mathematica* evolution has depended upon one person, Stephen Wolfram. Wolfram in 1979 began the construction of the modern computer algebra system for technical computing, that was essentially the zero-th version of *Mathematica*. The computer algebra system *Mathematica* (see [43]) is also written in C language. Despite the fact that this is a newer system, it already has about the same capabilities of symbolic computations as Macsyma. Unlike other computer algebra

systems, *Mathematica* has powerful means of computer graphics including 2D and 3D plots of functions, geometrical objects, contour, density plots, intersecting surfaces, illumination models, color PostScript output, animation, etc.

In *Mathematica* language, there exists a concept of basic primitive. For example, the expression (the basic computational primitive) can be used for representing mathematical formulas, lists, graphics; or the list (the basic data type primitive) can be used for representing as well many objects (vectors, matrices, tensors, etc.).

At first, *Mathematica* was used mainly in the physical sciences, engineering and mathematics. But over the years, it has become important in a wide range of fields, for instance, physical, biological, and social sciences, and other fields outside the sciences.

With the symbolic, numerical, and graphical calculus capabilities, *Maple* and *Mathematica* have become as a pair, one of the most powerful and complementary tools for students, professors, scientists, and engineers. Moreover, based on our experience of working with *Maple* and *Mathematica*, these systems are optimal from the point of view of requirements of memory resources and have outstanding capabilities for graphic display ([1], [10], [20], [11], [12], [41]).

The core of this book, is a large number of problems and their solutions that have been obtained simultaneously with the help of *Maple* and *Mathematica*. The structure of the book consists of the two parts.

In Part I, the foundations of the computer algebra systems *Maple* and *Mathematica* are discussed, starting from the introduction, basic concepts, elements of language (Chapters 1 and 2).

In Part II, the solutions of mathematical problems are described with the aid of the two computer algebra systems *Maple* and *Mathematica* and are presented in parallel, as in a dictionary. We consider a variety areas of mathematics that appeal to students such as algebra, geometry, calculus, complex functions, special functions, integral transforms, and mathematical equations (Chapters 3–10).

It should be noted that the results of the problems are not explicitly shown in the book (formulas, graphs, etc.). In music, a musician must be able to read musical notes and has developed the skill to follow the music from his studies. This skill allows him to read new musical notes and be capable of hearing most or all of the sounds (melodies, harmonies, etc.) in their head without having to play the music piece. By analogy, in Mathematics, we believe a scientist, engineer or mathematician

must be able to read and understand mathematical codes (e.g., *Maple*, *Mathematica*) in their head without having to execute the problem.

Following an approach based on solutions of problems rather than technical details, is, we believe, the ideal way for introducing students to mathematical and computer algebra methods. The text consists of core material for implementing *Maple* and *Mathematica* into different undergraduate mathematical courses concerning the areas mentioned above. Also, we believe that the book can be useful for graduate students, professors, and scientists and engineers. Any suggestions and comments related to this book are most appreciated. Please send your e-mail to [inna@fisica.uson.mx](mailto:inna@fisica.uson.mx) or [carlos@fisica.uson.mx](mailto:carlos@fisica.uson.mx).

We would like to express our gratitude to the Mexican Department of Public Education (SEP) and National Council for Science and Technology (CONACYT), for supporting this work under grant no. 43208. Also we would like to express our sincere gratitude to Profs. A. D. Polyanin and A. V. Manzhirov for their helpful ideas and commentaries. Finally, we wish to express our special thanks to Mr. Stephen Soehnlen and Mag. Wolfgang Dollhäubl from Springer Vienna for their invaluable and continuous support.

June 2007

Inna Shingareva  
Carlos Lizárraga-Celaya

# Contents

<b>Part I Foundations of <i>Maple</i> and <i>Mathematica</i></b>	<b>1</b>
<b>1 <i>Maple</i></b>	<b>3</b>
1.1 Introduction . . . . .	3
1.1.1 History . . . . .	3
1.1.2 Basic Features . . . . .	4
1.1.3 Design . . . . .	4
1.2 Basic Concepts . . . . .	5
1.2.1 First Steps . . . . .	5
1.2.2 Help System . . . . .	5
1.2.3 Worksheets and Interface . . . . .	5
1.2.4 Packages . . . . .	6
1.2.5 Numerical Evaluation . . . . .	7
1.3 <i>Maple</i> Language . . . . .	7
1.3.1 Basic Principles . . . . .	8
1.3.2 Constants . . . . .	10
1.3.3 Functions . . . . .	11
1.3.4 Procedures and Modules . . . . .	14
1.3.5 Control Structures . . . . .	16
1.3.6 Objects and Operations . . . . .	18
<b>2 <i>Mathematica</i></b>	<b>21</b>
2.1 Introduction . . . . .	21
2.1.1 History . . . . .	21
2.1.2 Basic Features . . . . .	22
2.1.3 Design . . . . .	22
2.2 Basic Concepts . . . . .	23
2.2.1 First Steps . . . . .	23
2.2.2 Help System . . . . .	23
2.2.3 Notebook and Front End . . . . .	24
2.2.4 Packages . . . . .	24
2.2.5 Numerical Evaluation . . . . .	25
2.3 <i>Mathematica</i> Language . . . . .	26
2.3.1 Basic Principles . . . . .	26
2.3.2 Constants . . . . .	32

2.3.3 Functions . . . . .	32
2.3.4 Modules . . . . .	36
2.3.5 Control Structures . . . . .	37
2.3.6 Objects and Operations . . . . .	39
<b>Part II Mathematics: <i>Maple</i> and <i>Mathematica</i></b>	<b>45</b>
<b>3 Graphics</b>	<b>47</b>
3.1 Simple Graphs . . . . .	47
3.2 Various Options . . . . .	48
3.3 Multiple Graphs . . . . .	50
3.4 Text in Graphs . . . . .	53
3.5 Special Graphs . . . . .	54
3.6 Animations . . . . .	60
<b>4 Algebra</b>	<b>65</b>
4.1 Polynomials . . . . .	65
4.2 Simplification . . . . .	67
4.3 Linear Algebra . . . . .	71
4.4 Tensors . . . . .	85
<b>5 Geometry</b>	<b>91</b>
5.1 Points in the Plane and Space . . . . .	91
5.2 Parametric Curves . . . . .	94
5.3 Implicitly Defined Curves . . . . .	96
5.4 Curves in Polar Coordinates . . . . .	96
5.5 Secant and Tangent Lines . . . . .	98
5.6 Tubes and Knots . . . . .	99
5.7 Surfaces in Space . . . . .	101
5.8 Level Curves and Surfaces . . . . .	102
5.9 Surfaces of Revolution . . . . .	104
5.10 Vector Fields . . . . .	105
5.11 Cylindrical Coordinates . . . . .	106
5.12 Spherical Coordinates . . . . .	106
5.13 Standard Geometric Shapes . . . . .	107
<b>6 Calculus</b>	<b>109</b>
6.1 Differential Calculus . . . . .	109
6.2 Integral Calculus . . . . .	115
6.3 Series . . . . .	120
6.4 Fourier Series . . . . .	124
6.5 Multivariate and Vector Calculus . . . . .	127

<b>7 Complex Functions</b>	<b>135</b>
7.1 Complex Algebra . . . . .	135
7.2 Functions, Limits, Continuity . . . . .	137
7.3 Sequences and Series . . . . .	140
7.4 Transformations and Mappings . . . . .	143
7.5 Differentiation and Integration . . . . .	144
7.6 Singularities . . . . .	145
<b>8 Special Functions</b>	<b>147</b>
8.1 Special Integrals . . . . .	147
8.2 Orthogonal Polynomials . . . . .	149
8.3 Trascendental Equations . . . . .	150
8.4 Differential Equations . . . . .	151
8.5 Infinite Series . . . . .	153
8.6 Distribution Functions . . . . .	154
<b>9 Integral and Discrete Transforms</b>	<b>155</b>
9.1 Laplace Transforms . . . . .	155
9.2 Integral Fourier Transforms . . . . .	161
9.3 Discrete Fourier Transforms . . . . .	164
9.4 Hankel Transforms . . . . .	168
<b>10 Mathematical Equations</b>	<b>171</b>
10.1 Algebraic and Trascendental Equations . . . . .	171
10.2 Ordinary Differential Equations . . . . .	177
10.3 Partial Differential Equations . . . . .	187
10.4 Integral Equations . . . . .	204
References . . . . .	231
General Index . . . . .	235
<i>Maple</i> Index . . . . .	244
<i>Mathematica</i> Index . . . . .	254

Part I

## Foundations of *Maple* and *Mathematica*

# Chapter 1

## *Maple*

### 1.1 Introduction

#### 1.1.1 History

We begin with a brief history of *Maple*, from a research project at a university to a leading position in education, research, and industry.

*The first concept* of *Maple* and initial versions were developed by the Symbolic Computation Group at the University of Waterloo in the early 1980s.

*In 1988*, the new Canadian company Waterloo Maple Inc., was created to commercialize the software. While the development of *Maple* was done mainly in research labs at Waterloo University and at the University of Western Ontario, with important contribution from worldwide research groups in other universities.

*In 1990*, the first graphical user interface was introduced for Windows in version V.

*In 2003*, a Java “standard” user interface was introduced in version 9.

*In 2005*, *Maple* version 10 comes with a “document mode” as part of the user interface.

*In 2007*, *Maple* version 11 is introduced, it comes with an improved smart document environment to facilitate the user-interface learning curve. This tool integrates in an optimal way, all different sources and types of related information to the problem that the user is solving in that moment. This newer version includes more mathematical tools for modeling and problem analysis.

### 1.1.2 Basic Features

*Fast symbolic, numerical* computation, and interactive visualization;

*Easy to use*, help can be found within the program or on the Internet;

*Extensibility*;

*Accessible* to large numbers of students and researchers;

*Available* for almost all operating systems (MS Windows, Linux, Unix, Mac OS);

*Powerful programming language*, intuitive syntax, easy debugging;

*Extensive library* of mathematical functions and specialized packages;

*Two forms* of interactive interfaces: a command-line and a graphic environment;

*Free resources*, (*Maple Application Center*), collaborative character of development (*Maple Community*);

*Understandable*, open-source software development path.

### 1.1.3 Design

*Maple* consists of three parts: the interface, the kernel (basic computational engine), and the library. *The interface* and *the kernel* form a smaller part of the system, which has been written in the C programming language; they are loaded when a *Maple* session is started.

*The interface* handles the input of mathematical expressions, output display, plotting of functions, and support of other user communication with the system. The interface medium is *the Maple worksheet*.

*The kernel* interprets the user input and carries out the basic algebraic operations, and deals with storage management.

*The library* consists of two parts: the basic library and a collection of packages. The basic library includes many functions in which resides most of the common mathematical knowledge of *Maple* and that has been coded in the *Maple* language.

## 1.2 Basic Concepts

### 1.2.1 First Steps

We type the *Maple* command to the right of the prompt symbol `>`, and at the end of the command we place a semicolon, and then press **Enter** (or **Shift+Enter** to continue the command onto the next line). *Maple* evaluates the command, displays the result, and inserts a new prompt.

```
> ?introduction
> evalf(gamma, 40);
> solve(11*x^3-9*x+17=0, x);
> sort(expand((y+1)^(10)));
> plot({4*sin(2*x), cos(2*x)^2}, x=0..2*Pi);
> plot3d(cos(x^2+y^2), x=0..Pi, y=0..Pi);
```

Above, the first line gives you introductory information about *Maple*, the second line returns a 40-digit approximation of Euler's constant  $\gamma$ , the third line solves the equation  $11x^3 - 9x + 17 = 0$  for  $x$ , the fourth line expresses  $(y + 1)^{10}$  in a polynomial form, the fifth line plots the functions  $4 \sin(2x)$  and  $\cos^2(2x)$  on the interval  $[0, 2\pi]$ , and the last line plots the function  $\cos(x^2 + y^2)$  on the rectangle  $[0, \pi] \times [0, \pi]$ .

### 1.2.2 Help System

*Maple* contains a complete online help system. You can use it to find information about a specific topic and to explain the available commands.

You can use:

- ?command, `help(command)`,
- the **Help** menu,
- by highlighting a command and then pressing F1
- (in newer versions of *Maple*  $\geq 9$ ).

### 1.2.3 Worksheets and Interface

*Maple worksheets* are files that keep track of a working process and organize it as a collection of expandable groups (see `?worksheet`, `?shortcut`), e.g.,

**Insert → Section**, insert a new group at the current level;  
**Insert → Subsection**, insert a subgroup;  
**Ctrl-k**, insert a new input group above the current position;  
**Ctrl-j**, insert a new input group below the current position;  
**Ctrl-t**, converts the current group into a text group;  
**Ctrl-r, Ctrl-g**, insert Standard Math typesetting;  
**Format → Convertto → StandardMath**, converts a selected formula into standard mathematical notation.

*Palettes* can be used for building or editing mathematical expressions without the need of remembering the *Maple* syntax  
(**View → Palettes → ShowAllPalettes**).

The *Maplet User Interface* (in versions of *Maple*  $\geq 8$ ) consists of *Maplet applications* that are collections of windows, dialogs, actions (see **?Maplets**).

#### 1.2.4 Packages

In addition to the standard library functions, a number of specialized functions are available in various packages (subpackages), in more detail, see **?index[package]**. A package (subpackage) function can be loaded in the form (see **?with**).

```
with(package); function(arguments);
with(package[subpackage]); function(arguments);
```

**Problem:** Define a square matrix ( $n=3$ ) of random entries and compute the determinant.

```
with(linalg): with(LinearAlgebra):
n := 3; A1 := randmatrix(n, n, entries=rand(-10..10));
B1 := det(A1);
A2:=RandomMatrix(n,n,generator=1..9,
                  outputoptions=[shape=triangular[upper]]);
B2 := Determinant(A2);
```

### 1.2.5 Numerical Evaluation

*Maple* gives an exact answer to arithmetic expressions with integers and reduces fractions. When the result is an irrational number, the output is returned in unevaluated form.

```
135+638; -13*77; 467/31; (3+8*4+9)/2; (-5)^(22); -5^(1/3);
```

*Most* computers represent both integer and floating point numbers internally using the binary number system. *Maple* represents the numbers in the decimal number system using the user-specified precision.

*Numerical* approximations:

`evalf(expr)`, numerical approximation of `expr` (to 10 significant digits),

`Digits`, global changing a user-specified precision, with the environment variable `Digits` (see `?Digits`, `?environment`),

`evalf(expr,n)`, local changing a user-specified precision, with the function `evalf(expr,n)`,

`evalhf(expr)`, numerical approximation of `expr` using a binary hardware floating point system.

```
evalf((-5)^(1/3)); evalf(sqrt(122)); evalf((1+sqrt(5))/2);
evalf(exp(1), 50); Digits := 50;      evalf(exp(1));
0.9;  evalf(0.9);  evalhf(0.9);
```

## 1.3 *Maple* Language

*Maple* language is a high-level programming language, well-structured, comprehensible. It supports a large collection of *data structures* or *Maple objects* (functions, sequences, sets, lists, arrays, tables, matrices, vectors, etc.) and operations among these objects (type-testing, selection, composition, etc.).

The *Maple* procedures in the library are available in readable form. The library can be complemented with locally user developed programs and packages.

### 1.3.1 Basic Principles

*Basic arithmetic operators:* `+ - * / ^`.

*Logic operators:* `and, or, xor, implies, not`.

*Relation operators:* `<, <=, >, >=, =, <>`.

*A variable name,* `var`, is a combination of letters, digits, or the underline symbol `(_)`, beginning with a letter, e.g., `a12_new`.

*Abbreviations* for the longer *Maple* functions or any expressions: `alias`, e.g., `alias(H=Heaviside); diff(H(t),t);` to remove this abbreviation, `alias(H=H);`

*Maple is case sensitive*, there is a difference between lowercase and uppercase letters, e.g., `evalf(Pi)` and `evalf(pi)`.

*Various reserved keywords,* symbols, names, and functions, these words cannot be used as variable names, e.g., operator keywords, additional language keywords, global names that start with `(_)` (see `?reserved, ?infnames, ?inifncts, ?names`).

*The assignment/unassignment operators:* a variable can be “free” (with no assigned value) or can be assigned any value (symbolic, numeric) by the assignment operators `a:=b` or `assign(a=b)`. To unassign (clear) an assigned variable:

`x:='x', evaln(x), or unassign('x').`

*The difference* between the operators `(:=)` and `(=)`. The operator `var:=expr` is used to assign `expr` to the variable `var`, and the operator `A=B` — to indicate equality (not assignment) between the left- and the right-hand sides (see `?rhs`), e.g.,

`Equation:=A=B; Equation; rhs(Equation); lhs(Equation);`

*Statements,* `stats`, are input instructions from the keyboard that are executed by *Maple* (e.g., `break, by, do, end, for, function, if, proc, restart, return, save, while`).

*The new worksheet* (or the new problem) it is best to begin with the statement `restart` for cleaning *Maple*’s memory. All examples and problems in the book assume that they begin with `restart`.

The range operator `(..)`, an expression of type `expr..expr`, e.g.,  
`a[i] $ i=1..9;`

The statement separators semicolon `(;)` and colon `(:)`. The result of a statement followed with a semicolon `(;)` will be displayed, and it will not be displayed if it is followed by a colon `(:)`, e.g.,

```
plot(sin(x),x=0..Pi); plot(sin(x),x=0..Pi):
```

An expression, `expr`, is a valid statement, and is formed as a combination of constants, variables, operators and functions.

Data types, every expression is represented as a tree structure in which each node (and leaf) has a particular data type. For the analysis of any node and branch, the functions `type`, `whattype`, `nops`, `op` can be used.

A boolean expression, `bexpr`, is formed with the logical operators and the relation operators.

An equation, `Eq`, is represented using the binary operator `=`, and has two operands, the left-hand side `lhs` and the right-hand side `rhs`.

Inequalities, `Ineq`, are represented using the relation operators and have two operands, the left-hand side `lhs` and the right-hand side `rhs`.

A string, `Str`, is a sequence of characters having no value other than itself, cannot be assigned to, and will always evaluate to itself. For instance, `x:="string";` and `sqrt(x);` is an invalid function. Names and strings can be used with `convert` and `printf`.

Incorrect response. If you get no response or an incorrect response you may have entered or executed the command incorrectly. Do correct the command or interrupt the computation (the stop button in the Tool Bar menu).

Types of brackets:

Parentheses `(expr)`, grouping expressions, `(x+9)*3`, the function arguments, `sin(x)`.

Square brackets `[expr]`, lists, `[a,b,c]`, vectors, matrices, arrays.

Curly brackets `{expr}`, sets, `{a,b,c}`.

*Types of quotes:*

Forward-quotes 'expr', to delay evaluation of expression,  
'x+9+1', to clear variables,  $x := 'x'$ ;

Back-quotes `expr', to form a symbol or a name,  
'the name := 7';  $k := 5$ ; print('the value of k is',  $k$ );

Double quotes "expr", to create strings, and a single double quote ", to delimit strings.

*Previous results* (during a session) can be referred with symbols % (the last result), %% (the next-to-last result), %...%,  $k$  times, (the  $k$ -th previous result), a+b;  $\%^2$ ; %% $\%^2$ ;

*Comments* can be included with the sharp sign # and all characters following it upto the end of a line. Also the text can be inserted with **Insert → Text**.

*Maple source code* can be viewed for most of the functions, general and specialized (package functions).

```
interface(verboseproc=2);
print(factor); print('plots/arrow');
```

### 1.3.2 Constants

*Types of numbers:* integer, rational, real, complex, root, e.g.,

```
-55, 5/6, 3.4, -2.3e4, Float(23,-45), 3-4*I, Complex(2/3,3),
RootOf(_Z^3-2,index=1).
```

*Predefined constants:* symbols for definitions of commonly used mathematical constants, true, false, gamma, Pi, I, infinity, Catalan, FAIL, exp(1) (see ?ininame, ?constants).

*An angle* symbolically has dimension 1. *Maple* knows many units of angle (see ?Units/angle), e.g., convert(30\*degrees,radians);
  
convert(30,units,degrees,radians);

*The packages* ScientificConstants and Units (in *Maple* versions  $\geq 7$ ) provide valuable tools for scientists and engineers in Physics and Chemistry (see ?ScientificConstants, ?Units).

### 1.3.3 Functions

*Active functions* (beginning with a lowercase letter) are used for computing;

*Inert functions* (beginning with a capital letter) are used for showing steps in the problem-solving process; e.g., `diff`, `Diff`, `int`, `Int`, `limit`, `Limit`.

*Two classes of functions:* the library functions (predefined functions) and user-defined functions.

*Predefined functions:* most of the well known functions are predefined by *Maple* and they are known to some *Maple* functions (`diff`, `evalc`, `evalf`, `expand`, `series`, `simplify`). In addition, numerous special functions are defined (see `?FunctionAdvisor`). We will discuss some of the more commonly used functions.

*Elementary transcendental functions:* the exponential function, the natural logarithm, the general logarithm, the common logarithm, the trigonometric and hyperbolic functions and their inverses.

<code>exp(x);</code>	<code>ln(x);</code>	<code>log[b](x);</code>	<code>log10(x);</code>
<code>sin(x);</code>	<code>cos(x);</code>	<code>tan(x);</code>	<code>cot(x);</code>
<code>sec(x);</code>	<code>csc(x);</code>	<code>sinh(x);</code>	<code>cosh(x);</code>
<code>tanh(x);</code>	<code>coth(x);</code>	<code>sech(x);</code>	<code>csch(x);</code>
<code>arcsin(x);</code>	<code>arccos(x);</code>	<code>arctan(x);</code>	<code>arccot(x);</code>
<code>arcsec(x);</code>	<code>arccsc(x);</code>	<code>arcsinh(x);</code>	<code>arccosh(x);</code>
<code>arctanh(x);</code>	<code>arccoth(x);</code>	<code>arcsech(x);</code>	<code>arccsch(x);</code>

```
evalf(ln(exp(1)^3)); evalf(tan(3*Pi/4));
evalf(arccos(1/2)); evalf(tanh(1));
```

*Other useful functions* (see `?inifcn`)

`max/min`, maximum/minimum values of a set of real values.

`round`, `floor`, `ceil`, `trunc`, `frac`, converting real numbers to nearby integers, and the fractional part or real numbers.

*User-defined functions:* the functional operator (see `?->`), e.g., functions of one or many variables  $f(x) = \text{expr}$ ,  $f(x_1, \dots, x_n) = \text{expr}$ , the vector functions of one or many variables,  $f(x) = (x_1, \dots, x_n)$ ,  $f(x_1, \dots, x_n) = \langle f_1(x_1, \dots, x_n), \dots, f_n(x_1, \dots, x_n) \rangle$ .

```
f:=x->expr;  f:=(x1,...,xn)->expr; f:=x->[x1,...,xn];
f:=(x1,...,xn)->[f1(x1,...,xn),...,fn(x1,...,xn)];
```

*Alternative definitions of functions:* `unapply` converts an expression to a function, and a procedure is defined with `proc`.

```
f := unapply(expr, x);      f := proc(x) expr end;
```

*Evaluation* at  $x = a$ ,  $\{x = a, y = b\}$ , etc.

```
f(a);    subs(x=a, f(x));      eval(f(x), x=a);
f(a,b);  subs(x=a,y=b,f(x,y)); eval(f(x,y), {x=a, y=b});
```

*Composition operator* `@`, e.g., the composition function  $(f \circ f \circ \dots \circ f)(x)$  ( $n$  times) or  $(f_1 \circ f_2 \circ \dots \circ f_n)(x)$ .

```
(f @@ n)(x);      (f1 @ f2 @f3 @ ... @fn)(x);
```

**Problem:** Define the function  $f(x, y) = 1 - \sin(x^2 + y^2)$  and evaluate  $f(1, 2)$ ,  $f(0, a)$ ,  $f(a^2 - b^2, b^2 - a^2)$ .

```
f:=(x,y)->1-sin(x^2+y^2);
evalf(f(1,2));  f(0,a);  simplify(f(a^2-b^2,b^2-a^2));
```

**Problem:** Define the vector function  $h(x, y) = \langle \cos(x - y), \sin(x - y) \rangle$  and calculate  $h(1, 2)$ ,  $h(\pi, -\pi)$ , and  $h(\cos(a^2), \cos(1 - a^2))$ .

```
h:=(x,y)->[cos(x-y), sin(x-y)];
evalf(h(1,2));  h(Pi,-Pi);
combine(h(cos(a^2),cos(1-a^2)), trig);
```

**Problem:** Graph the real roots of the equation  $x^3 + (a - 3)^3x^2 - a^2x + a^3 = 0$  for  $a \in [0, 1]$ .

```
Sol:=[solve(x^3+(a-3)^3*x^2-a^2*x+a^3=0, x)];
for i from 1 to 3 do
    R || i := unapply(Sol[i], a):
    print(plot(R || i(a), a = 0..1, numpoints=500));
od:
```

**Problem:** For the functions  $f(x) = x^2$  and  $h(x) = x + \sin x$  calculate the composition function  $(f \circ h \circ f)(x)$  and  $(f \circ f \circ f \circ f)(x)$ .

```
f:=x->x^2; h:=x->x+sin(x);
f(h(f(x)));      (f@h@f)(x);  f(f(f(f(x))));  (f@@4)(x);
```

*Piecewise continuous functions* can be defined with `piecewise` or as a procedure with the control statement `if`.

```
f:=x->piecewise(cond1,expr1,expr2);
g:=x->piecewise(cond1,expr1,cond2,expr2,expr3);
f:=proc(x) if cond1 then expr1 else expr2 fi: end:
g:=proc(x) if cond1 then expr1 elif cond2 then expr2
            else expr3 fi: end:
```

**Problem:** Define and graph the function  $g(x) = \begin{cases} 0, & |x| > 1 \\ 1-x, & 0 \leq x \leq 1 \\ 1+x, & -1 \leq x \leq 0 \end{cases}$  in  $[-3, 3]$ .

```
g:=x->piecewise(x>=-1 and x <=0,1+x,x>=0 and x<=1,1-x,0);
plot(g(x), x=-3..3);
g := proc(x) if x >= -1 and x <= 0 then 1+x
            elif x >= 0 and x <= 1 then 1-x
            else 0         fi:           end;
plot(g, -3..3);
```

**Problem:** Graph the periodic extension of  $f(x) = \begin{cases} x, & 0 \leq x < 1 \\ 1, & 1 \leq x < 2 \\ 3-x, & 2 \leq x < 3 \end{cases}$  in  $[0, 20]$ .

```
f := proc(x) if x >= 0 and x < 1 then x
           elif x >= 1 and x < 2 then 1
           elif x >= 2 and x < 3 then 3-x fi: end:
plot(f, 0..3);
g:=proc(x) local a,b;
a:=trunc(x) mod 3; b:=frac(x); f(a+b); end:
plot(g, 0..20);
```

### 1.3.4 Procedures and Modules

In *Maple* language there are two forms of modularity: *procedures* and *modules*.

A *procedure* (see `?procedure`) is a block of statements which one needs to use repeatedly. A procedure can be used to define a function (if the function is too complicated to write by using the arrow operator), to create a matrix, a graph, a logical value, etc.

```
proc(args) local v1; global v2; options ops;
           stats; end proc;
```

where `args` is a sequence of arguments, `v1` and `v2` are the names of local and global variables, `ops` are special options (see `?options`), and `stats` are statements that are realized inside the procedure.

**Problem:** Define a procedure `maximum` that calculate the maximum of two arguments  $x, y$ , and then calculate the maximum of  $(34/9, 674/689)$ .

```
Maximum := proc(x, y) if x>y then x else y fi end;
Maximum(34/9, 674/689);
```

**Problem:** In Lagrangian mechanics,  $N$ -degree of freedom holonomic systems are described by the Lagrange equations [39]:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i, \quad i = 1, \dots, N,$$

where  $L = T - P$  is the Lagrangian of the system,  $T$  and  $P$  are kinetic and potential energy, respectively,  $q_i$  and  $Q_i$  are, respectively, the generalized coordinates and forces (not arising from a potential), and

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} = \sum_{j=1}^N \left( \frac{\partial}{\partial q_j} \frac{\partial L}{\partial \dot{q}_i} \dot{q}_j + \frac{\partial}{\partial \dot{q}_j} \frac{\partial L}{\partial \dot{q}_i} \ddot{q}_j \right).$$

Let the generalized forces,  $Q_i(q_1, \dots, q_n; \dot{q}_1, \dots, \dot{q}_n)$ , the kinetic and potential energy,  $T(q_1, \dots, q_n; \dot{q}_1, \dots, \dot{q}_n)$  and  $P(q_1, \dots, q_n)$ , be given.

Construct a procedure for deriving the Lagrange equations. As an example, consider the motion of a double pendulum of mass  $m$  and length  $L$  in the vertical plane due to gravity. For the pendulum ( $N=2$ ), the generalized coordinates are angles,  $A$  and  $B$ , the generalized non potential forces are equal to zero.

```
Eq_Lagrange := proc(q, L, Q)
local i, j, N, dq, d2q, Lq, Ldq, dLdq, Eq; N:=nops(q);
for i from 1 to N do dq[i] := cat(q[i], ',''); od;
for i from 1 to N do d2q[i] := cat(q[i], '''''); od;
for i from 1 to N do
  Lq := diff(L, q[i]); Ldq := diff(L, dq[i]);
  dLdq := add(diff(Ldq, q[j])*dq[j]
    + diff(Ldq, dq[j])*d2q[j], j=1..N);
  Eq[i]:=dLdq-Lq-Q[i]; od;
RETURN(normal(convert(Eq, list))); end;
T:=m*x^2*'A'`^2+m*x^2*'A'*'B'*cos(A-B)+1/2*m*x^2*'B'`^2;
P:=- m*g*x*cos(A) - m*g*x*(cos(A)+cos(B));
q:=[A, B]; g := omega^2*x; Q := [0,0];
Eq_L:=Eq_Lagrange(q, T-P, Q); Eq_L1 := factor(Eq_L);
```

*Recursive procedures* are the procedures that call themselves until a condition is satisfied (see `?remember`).

**Problem:** Calculate Fibonacci numbers and the computation time for  $F(n) = F(n - 1) + F(n - 2)$ ,  $F(0) = 0$ ,  $F(1) = 1$ .

```
Fib := proc(n::integer) option remember;
    if n<=1 then n else Fib(n-1)+Fib(n-2) fi; end;
NF := NULL:
for i from 10 to 40 do
    NF:=NF, Fib(i); od: NF:=[NF];
ti := time(): Fib(3000); tt := time()-ti;
```

A *module* (see `?module`) is a generalization of the procedure concept.

Since the procedure groups a sequence of statements into a single statement (block of statements), the module groups related functions and data.

```
module() export v1; local v2; global v3; option ops;
           stats;   end module;
```

where `v1`, `v2`, and `v3` are the names of export, local, and global variables, respectively, `ops` are special options (see `?module[options]`), and `stats` are statements that are realized inside the module.

**Problem:** Define a module `maxmin` that calculates the maximum and the minimum of two arguments  $x, y$ , and then calculate the maximum and minimum of  $(34/9, 674/689)$ .

```
MaxMin := module() local x, y; export maximum, minimum;
           maximum:=(x,y)->if x>y then x else y fi;
           minimum:=(x,y)->if x<y then x else y fi;
end module;
MaxMin:=maximum(34/9, 674/689);
MaxMin:=minimum(34/9, 674/689);
```

### 1.3.5 Control Structures

In *Maple* language there are essentially *two control structures*: the selection structure `if` and the repetition structure `for`.

```

if cond1 then expr1 else expr2 fi;
if cond1 then expr1 elif cond2 then expr2
            else expr3 fi;
for i from i1 by step to i2 do stats od;
for i from i1 by step to i2 while cond1 do stats od;
for i in expr1 do stats od;
for i in expr1 while expr2 do stats od;

```

where `cond1` and `cond2` are conditions, `expr1`, `expr2` are expressions, `stats` are statements, `i`, `i1`, `i2` are, respectively, the loop variable, the initial and the last values of `i`. These operators can be nested. The operators `break`, `next`, `while` inside the loops are used for breaking out of a loop, to proceed directly to the next iteration, or for imposing an additional condition.

**Problem:** Define the function *double factorial* for any integer  $n$ ,

$$n!! = \begin{cases} n(n-2)(n-4) \dots (4)(2), & n = 2i, i = 1, 2, \dots, \\ n(n-1)(n-3) \dots (3)(1), & n = 2i + 1, i = 0, 1, \dots \end{cases}$$

```

N1 := 20; N2 := 41;
FD := proc(N) local P, i1, i; P := 1;
    if modp(n, 2) = 0 then i1:=2 else i1:=1 fi:
    for i from i1 by 2 to N do P:=P*i; od: end:
printf("%7.0f!! = %20.0f", N1, FD(N1));
printf("%7.0f!! = %20.0f", N2, FD(N2));

```

**Problem:** Calculate the values  $x_i$ , where  $x_i = (x_{i-1} + 1/x_{i-1})$ ,  $x_0 = 1$ , until  $|x_i - x_{i-1}| \leq \varepsilon$  ( $i = 1, 2, \dots, n$ ,  $n = 10$ ,  $\varepsilon = 10^{-3}$ ).

```

n:=10; x:=1; epsilon:=10^(-3);
for i from 1 to n do xp := x: x := evalf((x+1/x));
    if abs(x-xp) <= epsilon then break
    else
        printf(" x= %20.8f \n", x) fi:
od:

```

**Problem:** Find an integer  $N(x)$ ,  $x \in [a, b]$  such that  $\sum_{i=1}^{N(x)} i^{-x} \geq c$ , where  $a = 1$ ,  $b = 2$ , and  $c = 2$ .

```
a:=1; b:=2; c:=2;
for x from a to b by 0.01 do S := 0:
  for i from 1 to 100 while S < c do
    S := S + evalf(i^(-x));
  od:
  printf("x= %7.4f      N(x)= %7d \n", x, i);
od:
```

### 1.3.6 Objects and Operations

*Maple objects*, sequences, lists, sets, tables, arrays, vectors, matrices, are used for representing more complicated data.

```
Sequence1 := expr1, expr2, expr3, ..., exprn;
Sequence2 := seq(f(i), i=a..b);
List1 := [Sequence1]; Set1 := {Sequence1};
Table1 := table([expr1=A1, ..., exprN=AN]);
Array1 := array(n..m);
Vec1 := array(1..n, [a1,a1,...,an]);
Vec2 := Vector(<a1,a2,a3,...,an>);
M1:=array(1..n,1..m, [[a11,...,alm], ..., [an1,...,anm]] );
with(linalg): M2:=matrix(n,m, [a11,a12,...,anm]);
M3 := Matrix(<<a11,a21,a31> | <a12,a22,a32>>);
M4 := Matrix([[a11,a12],[a21,a22],[a31,a32]]);
```

*Sequences, lists, sets* are groups of expressions. *Maple* preserves the order and repetition in sequences and lists and does not preserves it in sets. The order in sets can change during a *Maple* session.

A *table* is a group of expressions represented in tabular form. Each entry has an index (an integer or any arbitrary expression) and a value (see `?table`).

An *array* is a table with integer range of indices (see `?array`). In *Maple* arrays can be of any dimension (depending of computer memory).

A *vector* is a one-dimensional array with positive range integer of indices (see `?vector`, `?Vector`).

A *matrix* is a two-dimensional array with positive range integer of indices (see `?matrix`, `?Matrix`).

```
S1 := x, y, z, a, b, c;
L1:=[1,sin(x),cos(x),sin(2*x),cos(2*x),sin(3*x),cos(3*x)];
Set1 := {x, y, z}; A1:= array(-1..3);
A2 := array(1..4, [1,2,3,4]); A3 := array([1,2,3,4]);
```

*Basic operations* with data structures:

Create the empty structures, `NULL`, `:=`, `[ ]`.

```
Seq1 := NULL; List1:=[ ]; List_2:=NULL; Set1:={};
Tab1 := table(); Array1 := array(-10..10);
Vec1 := vector(10); Matrix1 := matrix(10, 10);
```

Concatenate structures, `||`, `op`, `[ ]`, `cat`.

```
Seq3 := Seq1 || Seq2; Seq3 := cat(Seq1, Seq2);
List3 := [op(List1, List2)];
```

Extract an *i-th* element from a structure, `[ ]`, `op`, `select`, `has`.

```
List1[i];      op(i, List1);   Array1[i, j];
op(i, Set1);  select(has, Set1, element);
```

Determine the number of elements in a structure, `nops`.

```
nops(List1);  nops(Set1);
```

Create a substructure, `op`, `[ ]`.

```
List2 := [op(n1..n2, List1)];  List2 := List1[n1..n2];
```

where  $n_1 \leq n_2 \leq n$  and  $n$  is a number of elements of `List1`.

Replace the *i-th* element of a structure, `:=`, `[ ]`, `subsop`, `subs`, `evalm`.

```
List1[i] := val;  subsop(i=val, List1);
A1 := subs(A[i,j]=a+b, evalm(A));
```

*Insert an element or some elements into a structure, [ ], op.*

```
List2:=[op(List1), A1]; List2:=[A1, op(List1)];
List3:=[op(n1..n2,List1), A1, A2, A3, A4, op(List2)];
```

*Create a structure according to a formula or with some special properties, e.g., zero, identity, sparse, symmetric, diagonal, etc. (in more detail, see Sect. 4.3).*

```
List1 :=[seq(f(i),i=n..m)];
Matrix1:=matrix(2,2,(i,j)->i+j); Vector1:=vector(2,i->i^2);
with(LinearAlgebra):ZeroMatrix(3,3);IdentityMatrix(3,3);
```

*Operations with sets, matrices:* the union and intersection of sets, removing elements from sets, the sum, difference, multiplication, division, scalar multiplication of matrices, union, intersect, minus, remove, has, evalm, &\*.

```
Set3 := Set1 union Set2; Set3 := Set1 intersect Set2;
Set3 := Set1 minus Set2; Set2 := remove(has, Set1, A1);
Mat3 := evalm(M1 &* M2);
```

*Apply a function to each element of a structure, map.*

```
Set2 := map(func, Set1);
```

**Problem:** A sequence of numbers  $\{x_i\}$  ( $i = 0, \dots, N$ ) is defined by  $x_{i+1} = ax_i(1 + x_i)$  ( $0 < a < 10$ ), where  $a$  is a given parameter. Define a list of coordinates  $[i, x_i]$  such that  $a = 3$ ,  $x_0 = 0.1$ ,  $N = 100$ . Plot the graph of the sequence  $\{x_i\}$ .

```
a:=3; x[0]:=0.1; N:=100;
for i from 1 to N do x[i]:=a*x[i-1]*(1-x[i-1]): od:
Seq1 := seq([i, x[i]], i=2..N):
plot([Seq1],style=point,symbol=circle,symbolsize=5);
```

**Problem:** Observe the function behavior  $y(x) = \cos(6(x - a \sin x))$ ,  $x \in [-\pi, \pi]$ ,  $a \in [\frac{1}{2}, \frac{3}{2}]$ .

```
with(plots): y:=x->cos(6*(x-a*sin(x))); G:=NULL; N:=20;
for i from 0 to N do
  a:=1/2+i/N; G:=G,plot(y(x),x=-Pi..Pi); od:
G:=[G]: display(G,insequence=true);
```

# Chapter 2

## *Mathematica*

### 2.1 Introduction

#### 2.1.1 History

*In 1979–1981*, Stephen Wolfram constructed SMP (*Symbolic Manipulation Program*), the first modern computer algebra system (SMP was essentially Version Zero of *Mathematica*).

*In 1986–1988*, Stephen Wolfram developed the first version of *Mathematica*. The concept of *Mathematica* was a single system that could handle many specific problems (e.g., symbolic, numerical, algebraic, graphical). In 1987, Wolfram founded a company, *Wolfram Research*, which continues to extend *Mathematica*.

*In 1991*, the second version of *Mathematica* appears with more built-in functions, MathLink protocol for interprocess and network communication, sound support, notebook front end.

*In 1996*, the third version introduced interactive mathematical typesetting system, exporting HTML, hyperlinks, and many other functions.

*In 1999*, the fourth version of *Mathematica* appears with important enhancements in speed and efficiency in numerical calculation, publishing documents in a variety of formats, and enhancements to many built-in functions.

*In 2003*, in the fifth version of *Mathematica* the core coding was improved and the horizons of *Mathematica* are more extended (e.g., in numerical linear algebra, in numerical solutions for differential

equations, in supporting an extended range of import and export graphic and file structures, in solving equations and inequalities symbolically over different domains).

In 2007, version 6 is introduced. In this upgrade, *Mathematica* has been substantially redesigned in its internal architecture for better functionality. It increases interactivity, more adaptive visualization, ease of data integration, symbolic interface construction among others new features.

### 2.1.2 Basic Features

*Symbolic, numerical, acoustic, and graphical computations;*

*Extensibility, elegance;*

*Available* for MS Windows, Linux, UNIX, Mac OS operating systems;

*Powerful* and logical language;

*Extensive* library of mathematical functions and specialized packages;

*An interactive front end* with notebook interface;

*Interactive* mathematical typesetting system.

### 2.1.3 Design

*Mathematica* consists of two basic parts: *the kernel*, computational engine and the interface, *front end*. These two parts are separate, but communicate with each other via the *MathLink* protocol.

*The kernel* interprets the user input and performs all computations.

The kernel assigns the labels `In[number]` to the input expression and `Out[number]` to the output. These labels can be used for keeping the computation order. In the book, we will not include these labels in the examples. The result of kernels work can be viewed with the function `InputForm`.

```
Plot3D[Cos[x y],x,-Pi,Pi,y,-Pi,Pi]//InputForm
```

*The interface* between the user and the kernel is called *front end* and is used to display the input and the output generated by the kernel. The medium of the front end is the *Mathematica notebook*.

## 2.2 Basic Concepts

### 2.2.1 First Steps

We type *Mathematica* command and press the **RightEnter** key or **Shift+Enter** (or **Enter** to continue the command on the next line). *Mathematica* evaluates the command, displays the result, and inserts a horizontal line (for the next input).

```
?Arc*
N[EulerGamma, 40]
Solve[11*x^3-9*x+17==0, x]
Expand[(y+1)^10]
Plot[{4*Sin[2*x], Cos[2*x]^2}, {x, 0, 2*Pi}]
Plot3D[Cos[x^2+y^2], {x, 0, Pi}, {y, 0, Pi}]
```

The first line gives you information about *Mathematica*'s functions beginning with `Arc`; the second line returns a 40-digit approximation of Euler's constant  $\gamma$ ; the third line solves the equation  $11x^3 - 9x + 17 = 0$  for  $x$ ; the fourth line expresses  $(y + 1)^{10}$  in a polynomial form; the fifth line plots the functions  $4 \sin(2x)$  and  $\cos^2(2x)$  on the interval  $[0, 2\pi]$ ; and the last line plots the function  $\cos(x^2 + y^2)$  on the rectangle  $[0, \pi] \times [0, \pi]$ .

### 2.2.2 Help System

*Mathematica* contains many sources of online help:

*Mathematica* manual;

Help menu: the **HelpBrowser** (with functions grouped by topic), the **MasterIndex** (a complete alphabetical listing of all topics);

to type `?func` (information about a function), `??func` (for more extensive information), `Options[func]` (for information about options);

to mark a function and to press F1;

to use the symbols `(?, *)`, e.g., `?Inv*`, `?*Plot`, `?*our*`;

**Help** → **WhyTheBeep**, *Mathematica* will beep when you make an error.

### 2.2.3 Notebook and Front End

*Mathematica notebooks* are electronic documents that may contain *Mathematica* output, text, graphics (see ?Notebook). You can work simultaneously with many notebooks.

*Cells*: a *Mathematica* notebook consists of a list of cells. Cells are indicated along the right edge of the notebook by blue brackets. Cells can contain subcells, and so on. The kernel evaluates a notebook cell by cell.

*Operations with cells*: a horizontal line across the screen is the beginning of the new cell, to open (close) cells or to change their format click (double-click) on the cell brackets, e.g., for changing the background cell color, click on a cell bracket, then

`Format → BackgroundColor → Yellow.`

*Different types of cells*: input cells (for evaluation), text cells (for comments), Title, Subtitle, Section, Subsection, etc., can be found in the menu `Format → Style`.

*Palettes* can be used for building or editing mathematical expressions, texts, graphics and allows one to access by clicking the mouse to the most common mathematical symbols, e.g.,

`File → Palettes → BasicInput.`

### 2.2.4 Packages

In *Mathematica*, there exist many specialized functions and modules which are not loaded initially. They must be loaded separately from files in the *Mathematica* directory. These files are of the form `filename.m`. The information about packages can be found in the `MasterIndex` (see `Packages`, `StandardPackages`, `LoadingPackages`).

*The full name of a package* consists of a `context` and a `short` name, and it is written as `context`short`.

*Operations* with packages:

`<<context``, to load a package corresponding to a context;

`<<c1`c2``, to load a package corresponding to a sequence of contexts;  
`$Packages`, to display a list of loaded packages;  
`$Context`, to display the current context;  
`$ContextPath`, to get a list of contexts;  
`Names["context`*"]`, to get a list of the functions in a package.

**Problem:** Show the graphs of a Möbius strip centered around the  $z$ -axis with radii 5 and 2 and a map of North America.

```
<<Graphics` ; Names["Graphics`Shapes`"]
Show[Graphics3D[MoebiusStrip[5,2,50]]];
<<Miscellaneous`WorldPlot`
WorldPlot[NorthAmerica,AspectRatio->1];
```

### 2.2.5 Numerical Evaluation

*Mathematica* gives exact answers to arithmetic expressions with integers and reduces fractions. When the result is an irrational number, the output is returned in unevaluated form.

```
{135+638,-13*77,467/31,(3+8*4+9)/2,(-5)^(22),-5^(1/3)}
```

*Mathematica* represents the numbers in the decimal number system using a user-specified precision.

*Numerical* approximations:

`N[expr]`, `expr//N`, numerical approximation of `expr` (to 6 significant digits);  
`N[expr,n]`, `NumberForm[expr,n]`, numerical approximation of the expression to  $n$  significant digits;  
`ScientificForm[expr,n]`, `EngineeringForm[expr,n]`, scientific and engineering notation of numerical approximation of `expr` to  $n$  significant digits,

```
{Sqrt[17],Pi+2*Pi*I,1/3+1/5+1/7,N[E,30],N[Pi],Pi//N}
{ScientificForm[N[Sin[Exp[Pi]]],10],
EngineeringForm[N[Sin[Exp[Pi]]],10],
NumberForm[N[Sin[Exp[Pi]]],10]}
```

```
Compile[{x1,...},fb], Compile[{{x1,t1},{x2,t2}...},fb],
machine code of a function for improving the speed in numerical calculations (fb is the function body, and t1,t2,... are the types of variables, e.g., _Real, _Complex, _Integer).
```

```
n=0.5; f1=Compile[{x},Evaluate[(x^n*Exp[-x^n])^(1/n)]];  
Table[f1[x],{x,0.,50.,0.01}]//Timing  
f2[x_]:=(x^n*Exp[-x^n])^(1/n);  
Table[f2[x],{x,0.,50.,0.01}]//Timing
```

## 2.3 *Mathematica* Language

*Mathematica* language is a very powerful programming language based on systems of transformation rules, functional, procedural, and object-oriented programming techniques. This distinguishes it from traditional programming languages. It supports a large collection of *data structures* or *Mathematica objects* (functions, sequences, sets, lists, arrays, tables, matrices, vectors, etc.) and operations on these objects (type-testing, selection, composition, etc.). The library can be enlarged with custom programs and packages.

### 2.3.1 Basic Principles

*Symbol* in *Mathematica*, symb, refers to a symbol with the specified name, e.g., expressions, functions, objects, optional values, results, argument names.

A *name of symbol*, nam, is a combination of letters, digits, or certain special characters, not beginning with a digit, e.g., a12new. Once defined, a symbol retains its value until it is changed, cleared, or removed.

*Expression* is a symbol that represents an ordinary *Mathematica* expression in readable form, expr. The head of expr can be obtained with Head[expr]. The structure and various forms of expr can be analyzed with TreeForm, FullForm[expr], InputForm[expr], e.g.,

```
l1={5, 1/2, 9.1, 2+3*I, x, {A,B}, a+b, a*b}  
{Head /@ l1, FullForm[l1], InputForm[l1], TreeForm[l1]}
```

*Mathematica* is case sensitive, there is a difference between lowercase and uppercase letters, e.g., `Sin[Pi]` and `sin[Pi]` are different. All *Mathematica* functions begin with a capital letter. Some functions (e.g., `PlotPoints`) use more than one capital. To avoid conflicts, it is best to begin with a lower-case letter for all user-defined symbols.

The result of each calculation is displayed, but it can be suppressed by using a semicolon (`;`), e.g., `a=9; b=3; c=a*b`

```
Plot[Sin[x], x, 0, 2*Pi];
```

Basic arithmetic operators and the corresponding functions (in more detail, see `MasterIndex`, Operators):

<code>+</code>	<code>-</code>	<code>-</code>	<code>*</code>	<code>/</code>	<code>^</code>
<code>Plus</code>	<code>Subtract</code>	<code>Minus</code>	<code>Times</code>	<code>Divide</code>	<code>Power</code>

A missing symbol (`*`), `2a`, or a space, `a b`, also imply multiplication. So it is best to indicate explicitly all arithmetic operations.

```
Times[2, 3, 4, 5], Power[2, 3, 4].
```

Additional arithmetic operators and their equivalent functions:

<code>x++</code>	<code>x--</code>	<code>++x</code>	<code>--x</code>
<code>Increment</code>	<code>Decrement</code>	<code>PreIncrement</code>	<code>PreDecrement</code>
<code>x+=y</code>	<code>x-=y</code>	<code>x*=y</code>	<code>x/=y</code>
<code>AddTo</code>	<code>SubtractFrom</code>	<code>TimesBy</code>	<code>DivideBy</code>

In these operations  $x$  must have a numeric value, e.g., `x=5; x++;`

Logic and relation functions and their equivalent operators:

<code>And[x,y]</code>	<code>Or[x,y]</code>	<code>Xor[x,y]</code>	<code>Not[x,y]</code>	<code>Implies[x,y]</code>
<code>x&amp;&amp;y</code>	<code>x  y</code>		<code>!x</code>	
<code>Equal[x,y]</code>	<code>Unequal[x,y]</code>		<code>Less[x,y]</code>	<code>Greater[x,y]</code>
<code>x==y</code>	<code>x!=y</code>		<code>x&lt;y</code>	<code>x&gt;y</code>
<code>LessEqual[x,y]</code>			<code>GreaterEqual[x,y]</code>	
<code>x&lt;=y</code>			<code>x&gt;=y</code>	

Logic expressions can be compared using `LogicalExpand`.

*Patterns:* Mathematica language is based on pattern matching.

A *pattern* is an expression which contains an underscore character (\_).

The pattern can stand for any expression. Patterns can be constructed from the templates, e.g.,

$x_$	$x_ /; \text{cond}$	$\text{pattern}? \text{test}$	$x_ : \text{InitValue}$
$x^n_$	$x_n_$	$\{x_ , y_ \}$	$f[x_ ]$

**Problem:** Define the function  $f$  with any argument named  $x$ , define an expression satisfying a given condition.

```
f[x_]:=Abs[x]; f[t]
ex=t+Log[a]+Log[b]; {ex/.Log[b_]->Sin[b], ex/.Log[b]->Sin[b]}
```

*Basic transformation rules:*  $\rightarrow$ ,  $:>$ ,  $=$ ,  $:=$ ,  $^:=$ ,  $^=$

The rule `lhs->rhs` transforms `lhs` to `rhs`. Mathematica regards the left-hand side as a *pattern*.

The rule `lhs:>rhs` transforms `lhs` to `rhs`, evaluating `rhs` only after the rule is actually used.

```
a1=(a+b)^3; {a1/.Power[x_]->Power[Expand[x]], 
a1/.Power[x_]:>Power[Expand[x]]}
```

The assignment `lhs=rhs` (or `Set`) specifies that the rule `lhs->rhs` should be used whenever it applies.

The assignment `lhs:=rhs` (or `SetDelayed`) specifies that `lhs:>rhs` should be used whenever it applies, i.e., `lhs:=rhs` does not evaluate `rhs` immediately but leaves it unevaluated until the rule is actually called.

```
a1=TrigReduce[Sin[x]^2]; a2:=TrigReduce[Sin[x]^2]; {a1,a2}
x=a+b; {a1,a2}
```

Note that in many cases both the assignments produce identical results, but the operator `:=` must be used, e.g., for defining piecewise and recursive functions (see Subsect. 2.3.3).

The rule `lhs^:=rhs` assigns `rhs` to be the delayed value of `lhs`, and associates the assignment with symbols that occur at level one in `lhs`, e.g.,

```
D[int[f_,l_List],var_]:=int[D[f,var],l];
D[int[f[t],{t,0,n}],t]
Unprotect[D]; D[int[f_,l_List],var_]:=int[D[f,var],l];
Protect[D]; D[int[f[t],{t,0,n}],t]
```

The rule `lhs^=rhs` assigns `rhs` to be the value of `lhs`, and associates the assignment with symbols that occur at level one in `lhs`, e.g.,

```
Unprotect[{Cos,Sin}]; Cos[k_*Pi]:=(-1)^k; IntegerQ[k];
Sin[k_*Pi]:=0; IntegerQ[k]; Protect[{Cos, Sin}];
IntegerQ[n]^=True; {Cos[n*Pi], Sin[n*Pi]}
```

*Transformation rules* are useful for making substitutions without making the definitions permanent and are applied to an expression using the operator `/.` (`ReplaceAll`) or `//.` (`ReplaceRepeated`).

```
sol=Solve[x^2+4*x-10==0,x]; x+1/.sol
```

*Unassignment* of definitions:

`Clear[symb]` clears the symbol's definition and values, but does not clear its attributes, messages, or defaults.

`ClearAll[symb]` clears all definitions, values, attributes, messages, defaults.

`Remove[symb]` removes symbol completely.

`symb=.` clears a symbol's definition, e.g., `a=5;` `a=.`; `?a.`

To clear all global symbols defined in a *Mathematica* session can be produced by several ways, e.g.,

```
Clear["Global`*"]; ClearAll["Global`*"]; Remove["*"];
```

To recall a symbol's definition: `?symb`, e.g., `a=9;` `?a`

To recall a list of all global symbols that have been defined (during a session): `?`*`.

*Initialization:* in general, it is useful to start working with the following initialization (where `Off` turns off spelling error warnings):

```
ClearAll["Global`*"];
Off[General::spell, General::spell1]
```

*The difference between the operators (=) and (==):* the operator `lhs=rhs` is used to assign `rhs` to `lhs`, and the equality operator `lhs==rhs` indicates equality (not assignment) between `lhs` and `rhs`, e.g.,  
`{eq=A==B, eq, eq[[2]], eq[[1]]}`

*An expression,* `expr`, is a symbol that represents an ordinary *Mathematica* expression and there is a wide set of functions to work with it.

*Data types:* every expression is represented as a tree structure in which each node (and leaf) has a particular data type. For the analysis of any node and branch can be used a variety number of functions, e.g., `Length`, `Part`, a group of functions ending in the letter `Q` (`DigitQ`, `IntegerQ`, etc.). For example, the function `SameQ` or its equivalent `lhs==rhs` yields `True` if `lhs` is identical to `rhs`, and yields `False` otherwise.

*A boolean expression,* `bexpr`, is formed with the *logical operators* and the relation operators, e.g.,

```
LogicalExpand[!(p&&q)==>LogicalExpand[!p||!q]]
```

*An equation,* `eq`, is represented using the binary operator `==`, and has two operands, the left-hand side `lhs` and the right-hand side `rhs`.

*Inequalities,* `ineq`, are represented using the relational operators and have two operands, the left-hand side `lhs` and the right-hand side `rhs`.

*Strings:* a string is a sequence of characters having no value other than itself and can be used as labels for graphs, tables, and other displays. The strings are enclosed within double-quotes, e.g., `"abc"`.

*Some useful string manipulation functions:*

`StringLength[str]`, a number of characters in `str`;

`StringJoin[str1, ...]` or `str1 <> ...`, concatenation of strings;  
`StringReverse[str]` reverses the characters in `str`;  
`StringDrop[str, n, m]` eliminates characters in `str` from `n` to `m`;  
`StringTake[str, n, m]` takes characters in `str` from `n` to `m`;  
`StringInsert[str1, str2, n1, n2, ...]` inserts `str2` at each of the positions `n1, n2, ...` of `str1`;  
`StringReplace[str, str1 -> nstr1, ...]` replaces `str1` by `nstr1`;  
`StringPosition[str, substr]`, a list of the start and end positions of substring.

*Types* of brackets:

Parentheses (`expr`): grouping, `(x+9)*3`;  
Square brackets [`expr`]: function arguments, `Sin[x]`;  
Curly brackets {`expr`} : lists, `{a, b, c}`.

*Types* of quotes:

*Back-quotes* ‘`expr`’:

`fullname=context`short`, ‘ is a context mark;  
‘`x`’ is a format string character, `StringForm["`1`", `2`", a, b]`;  
‘ is a number mark, `12.8``, machine precision approximate number;  
‘ is a precision mark, `12.8`10`, arbitrary precision number with precision 10;  
‘‘ is an accuracy mark, `12.8``15`, arbitrary precision number with accuracy 15;

*Double-quotes* " `expr` " : to create strings;

*Previous results* (during a session) can be used with symbols % (the last result), %% (the next-to-last result), and so on.

*Comments* can be included within the characters (\*`comments`\*).

*Function application*: `expr//func` is equivalent to `fun[expr]`.

*Incorrect response:* if some functions take an “infinite” computation time, you may have entered or executed the command incorrectly. To terminate a computation, you can use:

Kernel → AbortEvaluation, Kernel → InterruptEvaluation,  
 Kernel → QuitKernel → Local.

### 2.3.2 Constants

*Types of numbers:* integer, rational, real, complex, root, e.g.,

```
{-5, 5/6, -2.3^-4, ScientificForm[-2.3^-4], 3-4*I},  
Root[#1^2 + #1 + 1 &, 2]
```

*Mathematical constants:* symbols for definitions of commonly used mathematical constants, e.g., Catalan, Degree, E, EulerGamma, I, Pi, Infinity, GoldenRatio (see MasterIndex, MathematicalConstants), e.g., {60Degree//N, N[E,30]}.

*Scientific constants:* valuable tools for scientists and engineers in Physics and Chemistry (see MasterIndex) can be applied with the packages: Miscellaneous‘Units‘, Miscellaneous‘PhysicalConstants‘, Miscellaneous‘ChemicalElements‘.

### 2.3.3 Functions

*Two classes of functions:* pure functions and functions defined in terms of a variable (*predefined* and *user-defined* functions).

*Pure functions* are defined without a reference to any specific variable. The arguments are labeled #1, #2, . . . , and an ampersand & is used at the end of definition.

```
f := Sin[#1]&; g := Sin[#1^2+#2^2]&;  
{f[x], f[Pi], g[x,y], g[Pi,Pi]}
```

*Predefined functions.* Most of the mathematical functions are predefined (see MasterIndex, MathematicalFunctions).

*Special functions.* Mathematica includes all the common special functions of mathematical physics (see MasterIndex, SpecialFunctions). We will discuss some of the more commonly used functions.

The names of mathematical functions are complete English words or the traditional abbreviations (for a few very common functions), e.g., `Conjugate`, `Mod`. Person's name mathematical functions have names of the form `PersonSymbol`, for example, the Legendre polynomials  $P_n(x)$ , `LegendreP[n,x]`.

*Elementary trascendental* functions:

`Exp[x]`, the exponential function;

`Log[x]`, `Log[b,x]`, the natural logarithm and the general logarithm;

`Sin`, `Cos`, `Tan`, `Cot`, `Sec`, `Csc`, the trigonometric functions;

`Sinh`, `Cosh`, `Tanh`, `Coth`, `Sech`, `Csch`, the hyperbolic functions;

`ArcSin`, `ArcSinh`, `ArcCos`, `ArcCosh`, ... (see `?Arc*`), the inverse trigonometric and hyperbolic functions.

`{Sin[30 Degree], Sin[Pi/3], Exp[5]/N}`

*Other useful* functions:

`Abs[x]`, `Sign[x]`, the absolute value and the sign functions;

`n!`, `Prime[n]`, `Fibonacci[n]`, the factorial, the  $n$ -th prime, and the Fibonacci functions;

`Round`, `Floor`, `Ceiling`, `IntegerPart`, `FractionalPart`, converting real numbers to nearby integers, and the fractional part of real numbers;

`Quotient`, `Mod`, `GCD`, `LCM`, the quotient and the remainder (in the Division Algorithm), the greatest common divisor and the least common multiple;

`Random[]`, `Random[type,range]`, the random functions;

`Timing`, the kernel computation time of the expression;

`IntegerQ`, `PolynomialQ`, ... (see `?*Q`), a group of functions ending in the letter `Q` can be used for testing for certain conditions and return a value `True` or `False`;

`Print`, this function can be used in loops, strings, and other displays.

```
{Quotient[25,3], Mod[25,3], Random[Real,{2,9},10]}
{Fibonacci[10000]//Timing,
 PolynomialQ[x^3*y+Sqrt[y]*y,y]}
```

*User-defined functions* are defined using the pattern  $x_-$ , e.g., the functions of one or  $n$  variables  $f(x) = \text{expr}$ ,  $f(x_1, \dots, x_n) = \text{expr}$ , the vector functions of one or  $n$  variables.

```
f[x_]:=expr; f=Function[x,expr];
f[x1_,...,xn_]:=expr; f[x_]:={x1,...xn};
f[x1_,...,xn_]:={f1[x1_],...,fn[x1_],...,xn_}};
```

*Evaluation* of a function or an expression without assigning a value can be performed using the replacement operator  $/.$ ,

```
f[a]    f[a,b]    expr /. x->a      expr /. {x->a,x->b}
```

*Composition functions* are defined with the operation `Composition` (the arguments of this operation are pure functions  $f_1, f_2, \dots$ ) and using the functions `Nest`, `NestList`, e.g., the composition function  $(f_1 \circ f_2 \circ \dots \circ f_n)(x)$  or  $(f \circ f \circ \dots \circ f)(x)$  ( $n$  times).

```
Composition[f1,f2,...]; Composition[f,...,f]; f1@f2@...
Nest[f,expr,n];          NestList[f,expr,n]   f@f@f...
```

**Problem:** Define the function  $f(x, y) = 1 - \sin(x^2 + y^2)$  and evaluate  $f(1, 2)$ ,  $f(0, a)$ ,  $f(a, b)$ .

```
f[x_,y_]:=1-Sin[x^2+y^2];
{N[f[1,2]], f[0,a], Simplify[f[a,b]]}
f1=Function[{x,y},1-Sin[x^2+y^2]];
{N[f1[1,2]], f1[0,a], Simplify[f1[a,b]]}
```

**Problem:** Define the vector function  $h(x, y) = \langle \cos(x - y), \sin(x - y) \rangle$  and calculate  $h(1, 2)$ ,  $h(\pi, -\pi)$ , and  $h(\cos(a^2), \cos(1 - a^2))$ .

```
h[x_,y_]:={Cos[x-y],Sin[x-y]};
{N[h[1,2]],h[Pi,-Pi],h[Cos[a^2],Cos[1-a^2]]//FullSimplify}
```

**Problem:** For the functions  $f(x) = x^2$  and  $h(x) = x + \sin x$  calculate the composition functions  $(f \circ h \circ f)(x)$  and  $(f \circ f \circ f \circ f)(x)$ .

```
f[x_]:=x^2; fF:=#1^2&; hF:=#1+Sin[#1]&;
{fF@hF@fF[x],fF@fF@fF[x],Nest[f,x,4],NestList[f,x,4]}
```

*Piecewise continuous functions* can be defined using the conditional operator /; or the functions Piecewise and UnitStep.

```
f[x_]:=expr/;cond f[x_]:=UnitStep[x-a]*UnitStep[b-x];
f[x_]:=Piecewise[{{cond1, val1}, {cond2, val2}, ...}]
```

**Problem:** Define and graph the function  $f(x) = \begin{cases} 0, & |x| > 1 \\ 1-x, & 0 \leq x \leq 1 \\ 1+x, & -1 \leq x \leq 0 \end{cases}$  in  $[-3, 3]$ .

```
f[x_]:=0/;Abs[x]>1; f[x_]:=1-x/;0<=x<=1;
f[x_]:=1+x/;-1<=x<=0; Plot[f[x], {x, -3, 3}];
```

**Problem:** Graph the periodic extension of  $f(x) = \begin{cases} x, & 0 \leq x < 1 \\ 1, & 1 \leq x < 2 \\ 3-x, & 2 \leq x < 3 \end{cases}$  in  $[0, 20]$ .

```
f[x_]:=x/;0<=x<1; f[x_]:=1/;1<=x<2; f[x_]:=3-x/;2<=x<=3;
f[x_]:=f[x-3]/; x>3; Plot[f[x], {x, 0, 20}];
```

*Recursive functions* are the functions that are defined in terms of themselves.

**Problem:** Calculate Fibonacci numbers,  $F(n) = F(n - 1) + F(n - 2)$ ,  $F(0) = 0$ ,  $F(1) = 1$ , using the memory feature and the analytic function. Compare the computation time in both cases.

```
$RecursionLimit=Infinity;
Fib[0]=0; Fib[1]=1; Fib[n_]:=Fib[n]=Fib[n-1]+Fib[n-2];
Table[Fib[i],{i,10,40}]
F[n_]:=(((1+Sqrt[5])/2)^n-((1-Sqrt[5])/2)^n)/Sqrt[5]
Fib[3000]//Timing
Expand[F[3000]]//Timing
```

### 2.3.4 Modules

A *module* is a local object that consists of several functions which one needs to use repeatedly (see `?Module`). A module can be used to define a function (if the function is too complicated to write by using the notation `f[x_]:=expr`), to create a matrix, a graph, a logical value, etc.

`Block` is similar to `Module`, the principal difference between them is that `Block` treats *the values* assigned to symbols as local, but *the names* as global, whereas `Module` treats *the names* of local variables as local.

`With` is similar to `Module`, the principal difference between them is that `With` uses *local constants* that are evaluated only once, but `Module` uses *local variables* whose values may change many time.

```
Module[{var1,...},body]; Module[{var1=val1,...},body];
Block[{var1,...},expr]; Block[{var1=val1,...},expr];
With[{var1=val1,var2=val2,...},expr];
```

where `var1,...` are local variables, `val1,...` are initial values of local variables, `body` is the body of the module (as a sequence of statements separated by semicolons). The final result of the module is the result of the last statement (without a semicolon). Also `Return[expr]` can be used to return an expression.

**Problem:** Define a module and module function, `mF`, that calculate the maximum of  $x$  and  $y$ , and then find the maximum of the values (34/9, 674/689).

```
Module[{x=34/9,y=674/689},If[x>y,x,y]]
mF[x_,y_]:=Module[{m},If[x>y,m=x,m=y]]; mF[34/9,674/689]
```

**Problem:** Compare the following calculations using `With`, `Module`, and `Block`.

```
f1[x_]:=With[{a=2,b=1},Tan[a*x]-b]; {f1[x],{a,b}}
f2[x_]:=Module[{i},Sum[x^(-2*i),{i,1,9}]];
f3[x_]:= Block[{i},Sum[x^(-2*i),{i,1,9}]];
{f2[4], f2[x], f2[i], f3[4], f3[x], f3[i]}
```

**Problem:** Consider a 2-degree of freedom holonomic system describing by the Lagrange equations, namely, the motion of a double pendulum of mass  $m$  and length  $L$  in the vertical plane due to gravity (see Subsect. 1.3.4). Construct a module for deriving the Lagrange equations.

```
eqLagrange[q_, L_, Q_]:=Module[{Lq,Ldq,Ldq1,dLdq,dLdq1,eq},
  L1=L/.{dq[[1]]->"A' ",dq[[2]]->"B' "}; n=Length[q];
  Lq=Map[D[L1,#1]&,q]/>{"A' "->dq[[1]],"B' "->dq[[2]]};
  Ldq=Map[D[L,#1]&,dq];
  Ldq1=Ldq/.{dq[[1]]->"A' ",dq[[2]]->"B' "};
  dLdq=Sum[Map[D[#1,q[[j]]]&,Ldq1]*dq[[j]]+
    Map[D[#1,dq[[j]]]&,Ldq]*d2q[[j]],{j,1,n}];
  dLdq1=dLdq/.{"A' "->dq[[1]],"B' "->dq[[2]]};
  eq=Factor[dLdq1-Lq-Q]
q={A,B}; Q={0,0}; dq=Map[#1'&,q]; d2q=Map[#1'&,q];
T=m*x^2*dq[[1]]^2+m*x^2*dq[[1]]*dq[[2]]*Cos[q[[1]]-q[[2]]]
  +1/2*m*x^2*dq[[2]]^2;
P=-m*g*x*Cos[q[[1]]]-m*g*x*(Cos[q[[1]]]+Cos[q[[2]]]);
g=omega^2*x; L=T-P; eqLagrange[q,L,Q]
```

### 2.3.5 Control Structures

In *Mathematica language* there are the following *two control structures*: the selection structures *If*, *Which*, *Switch* and the repetition structures *Do*, *While*, *For*.

```
If[cond, exprTrue]; If[cond, exprTrue, exprFalse];
If[cond, exprTrue, exprFalse, exprNeither];
Which[cond1,expr1,cond2,expr2,...];
Switch[expr,patt1,val1,patt2,val2,...];
Do[expr,{i,i1,i2,iStep}];
Do[expr,{i,i1,i2,iStep},{j,j1,j2,jStep},...];
While[cond, expr]; For[i=i1, cond, step, expr];
```

where *exprTrue*, *exprFalse*, *exprNeither* are expressions that execute, respectively, if the condition *cond* is *True*, *False*, and is neither *True* or *False*; *i*,*i1*,*i2* (*j*,*j1*,*j2*) are the loop variable and the initial and the last values of *i* (*j*).

The result of *Which* is the expression *expr1,expr2,...* corresponding to the first true condition *cond1,cond2,...*.

In `Switch`, the expression `expr` is compared with patterns `patt1`,... until a match is found and the corresponding value `val1`,... is the result. These operators can be nested.

The operators `Break[]`, `Continue[]`, `Goto[name]` inside the loops are used for breaking out of a loop, to proceed directly to the next iteration, or for transferring the control to the point `Label[name]`.

**Problem:** Compare the following calculations using `Which` and `Switch`.

```
f1[x_]:=Which[x>0,1,x==0,1/2,x<0,-1];
Plot[f1[x],{x,-2,2},PlotStyle->{Blue}]; f1[0]
f2[x_]:=Switch[x,_Integer,x,_List,Join[x,{c}],_Real,N[x]];
{f2[4], f2[Sum[2.3+0.01*i,{i,1,10}]], f2[{a,b}]}
```

**Problem:** Find all numbers from 1 to 30 which are not multiples of 2 or 5.

```
Do[If[Mod[i,2]==0||Mod[i,5]==0,,Print[i]],{i,1,30}]
```

**Problem:** Calculate  $20!$  using `Do`, `While`, `For` loops.

```
fact=1; n=20; Do[fact=fact*i, {i,1,n}]; fact
fact=1; n=20; While[i>0, fact=fact*i; i--]; fact
For[fact=1; i=1, i<=20, i++, fact=fact*i]; fact
```

**Problem:** Define the function *double factorial* for any integer  $n$ ,

$$n!! = \begin{cases} n(n-2)(n-4) \dots (4)(2), & n = 2i, i = 1, 2, \dots, \\ n(n-1)(n-3) \dots (3)(1), & n = 2i + 1, i = 0, 1, \dots \end{cases}$$

```
doubleFactorial[n_]:=Module[{p=1,i1},
  If[Mod[n,2]==0,i1=2,i1=1]; Do[p=p*i,{i,i1,n,2}];p]
n1=20; n2=41; {doubleFactorial[20],doubleFactorial[41]}
Print["20!!=", doubleFactorial[n1]]
Print["41!!=", doubleFactorial[n2]]
```

**Problem:** Calculate the values  $x_i$ , where  $x_i = (x_{i-1} + 1/x_{i-1})$ ,  $x_0 = 1$ , until  $|x_i - x_{i-1}| \leq \varepsilon$  ( $i = 1, 2, \dots, n$ ,  $n = 10$ ,  $\varepsilon = 10^{-3}$ ).

```
n=10; x=1; xp=1; epsilon=10^(-3); i=1;
While[Abs[x-xp]<epsilon||i<=n, xp=x; x=N[x+1/x];
      Print["x=",x]; i++]
```

**Problem:** Find an integer  $N(x)$ ,  $x \in [a, b]$  such that  $\sum_{i=1}^{N(x)} i^{-x} \geq c$ , where  $a = 1$ ,  $b = 2$ , and  $c = 2$ .

```
a=1; b=2; c=2;
Do[s=0; i=1; x=k;
  While[s<c && i<=100, s=s+N[i^(-x)]; i++];
  Print["s=",s," ", "x=",x," ", "N(x)=",i], {k,a,b,0.01}]
```

### 2.3.6 Objects and Operations

*Lists* are the fundamental objects in *Mathematica*. The other objects (e.g., sets, matrices, tables, vectors, arrays, tensors, objects containing data of mixed type) are represented as lists. A list is an ordered set of objects separated by commas and enclosed in curly braces, `{elements}`, or defined with the function `List[elements]`.

*Nested lists* are lists that contain other lists. There are many functions which manipulate lists and here we review some of the most basic.

*Basic manipulation functions:*

*Create empty lists:* `{ }`, `List[]`.

*Listable function.* Many *Mathematica* functions performed on a list will be performed on each element of the list.

```
list1:=1,2,3,4,5; Sqrt[list1]
```

*The basic standard operations* should be applied to lists with the same number of elements.

```
list1:=1,3,5,7; list2:=2,4,6,8; list1/list2
```

*Create lists* and nested lists:

`Range[n,m,step]`, lists of numbers;

`Table[expr,i,m,n,step]`, lists according to a formula;

`Array[f,n,nIni]`, list elements are functions `f[n]`;

`Table[expr,{i,m,n},{j,k,1}]`, nested lists according to a formula;

`Array[f,{m,n},{mIni,nIni}]`, nested lists with elements  $f[i,j]$ ;  
`Characters[str]`, `CharacterRange["c1","c2"]`, lists of characters.

```
f[x_]:=x^3-x^2-x-1; g[x_,y_]:=x^2+y^2;
{Range[0,Pi,Pi/3], Table[Sqrt[i],{i,1,20,4}],
 Array[f,20,0], Table[Sqrt[i+j],{i,1,4},{j,1,7}],
 Array[g,{3,5},{0,0}]}
{Characters["Maple&Mathematica"],
 CharacterRange["c","l"]}
```

*Determine the structure of lists:*

```
Length[list], Dimensions[list], TensorRank[list].

vec={a,b,c}; mat={{a,b,c},{d,e,f}};
tens={{{a,b,c},{d,e,f}},{{g,h,i},{j,k,l}}};
l1={vec,mat,tens};
{Length/@l1,Dimensions/@l1,TensorRank/@l1}
```

*Extract an  $i$ -th element from a list or a nested list:*

`Part[list,i]` or `list[[i]]`, `First[list]`, `Last[list]`, the first and the last elements of `list`.

```
list1=Range[1,20,3]; list2 =Table[(i+j)^2,{i,1,4},{j,1,7}];
{Part[list1, 4], list1[[4]], list2[[2,4]]}
```

*Create a substructure (a part of a list):*

`Rest[list]`, a list without the first element of `list`;  
`Delete[list,n]`, a list without the element in the  $n$ -th position of `list`;  
`Take[list,{m,n}]`, a list with the elements taken within the range;  
`Drop[list,{m,n}]`, a list with the elements deleted within the range;  
`Select[list,criterion]`, a list with the elements satisfying a criterion;

`Cases[list, pattern]`, a list with the elements matching a pattern;

`DeleteCases[list, pattern]`, a list with the removed elements matching a pattern.

```
f1[x_]:=Sqrt[x^2-x-1]; list1=Array[f1,10,0];
{Take[list1,{2,7}], Drop[list1,{2,7}]}
Select[{0,1,-2,Pi,a}, #>0&]
Cases[{1,-2,I,"a",b}, x_?StringQ]
DeleteCases[{1,-2,I,"a",b}, x_?StringQ]
```

*Insert* an element into a list:

`Append[list, x]` or `Prepend[list, x]`, insert an element `x` to the right or to the left of the last element of `list`;

`Insert[list, x, n]`, insert an element `x` in position `n`.

```
list1=Table[i^2,{i,1,20,2}];
{Append[list1,a], Insert[list1, a, 5]}
```

*Replace* the  $n$ -th element of a list by  $x$ , `ReplacePart[list, x, n]`.

```
list1=CharacterRange["0", "9"]; ReplacePart[list1,a,5]
```

*Rearrange* lists:

`Sort[list]` and `Reverse[list]`, sorting and reversing of lists;

`RotateLeft[list, n]`, `RotateRight[list, n]`, cycling of lists.

```
list1=Table[Random[Integer,{1,20}],{i,1,10}]
{Sort[list1],RotateLeft[list1,2],RotateRight[list1,-2]}
```

*Concatenation of lists:* `Join[list1, list2, ...]`

```
Join[a,b,c,d,e,f]
```

*Manipulation* with nested lists:

`TreeForm[list]`, visualization of nested lists as a tree;

`Depth[list]`, the number of levels in a nested list;

`Level[list, levels]`, a list of all sublists of `list` on levels;

`Level[list,levels,f]`, applies the function `f` to the sequence of sublists;

`Flatten[list]`, convert a nested list into a simple list;

`Flatten[list,n]`, partial flattening to level  $n$ ;

`Partition[list, n]`, converts a simple list into sublists of length  $n$ .

```
tensor={{a,b,c},{d,e,f}},{g,h,i},{j,k,l}};
TreeForm[tensor]
list1={{a1,a2,a3},{a4,a5,a6}}; list2=Flatten[list1];
{Depth[list1], Level[list1,2], Level[list1,1],
Level[list1,{1},Subtract], Partition[list2, 2]}
```

*Sets* are represented as lists:

`Union[list1, ...]`, a list of the distinct elements of lists;

`Intersection[list1,...]`, intersection of lists;

`Subset[list]`, a list of all subsets of the elements in `list`. The package `DiscreteMath`Combinatorica`` includes a number useful set functions.

*Vectors* are represented as lists, vectors are simple lists. Vectors can be expressed as single columns with `ColumnForm[list, horiz, vert]`.

```
{vec={a,b,c}, Length[vec], Dimensions[vec],
TensorRank[vec],VectorQ[vec]}
a[i_]=1/i; v1=N[Array[a,5]]; v1//ColumnForm[v1,Right]
b[i_]=1/i^2; v2=N[Array[b,5]]; v2//ColumnForm[v2,Right]
{a*v1+b*v2, v1.v2}
```

*Tables, matrices, and tensors* are represented as nested lists. There is no difference between the way they are stored: they can be generated using the functions `MatrixForm[list]`, `TableForm[list]`, or using the nested list functions (see above). Matrices and tables can also be conveniently generated using the *palette*,

`Input → CreateTable/Matrix → Palette`.

```
A1 = {{a1, a2}, {a3, a4}}; {MatrixForm[A1],TableForm[A1]}
```

*Basic manipulation functions* with matrices, tables, and tensors:

A *matrix* is a list of vectors. Matrices can be combined using the operations: addition (+), subtraction (-), scalar (\*) and matrix multiplication (.), in more detail about matrix manipulations, see Sect. 4.3.

```
m1={{a1,a2},{a3,a4}}; m2={{b1,b2},{b3,b4}};
{MatrixForm[m1], MatrixForm[m2]}
{4*m1-m2//MatrixForm, m1.m2//MatrixForm}
```

*Some useful functions* for tables:

`TableForm[list, options]`, generating tables with some properties;

`PaddedForm[list,{n,m}]`, formatting numerical tables.

```
t1={{a1,2*a2,a3},{-a4,a5,-a6}}; TableForm[t1]
TableForm[t1,TableAlignments->Right, TableHeadings->
 {"r1","r2"}, {"c1","c2","c3"}},TableSpacing->{3,3}]
t2=Table[{i,N[Sin[i]],N[Cos[i]]},{i,1,5}]; TableForm[t2]
f1:=PaddedForm[i,3]; f2:=PaddedForm[N[Sin[i]],[12,5]];
t3=Table[{f1,f2},{i,1,5}]; TableForm[t3]
```

A *tensor* is a list of matrices with the same dimensionality (in more detail about tensor manipulations, see Sect. 4.4).

*Some useful functions* for tensors:

`Table, Array`, creating tensors;

`TreeForm, MatrixForm`, visualizing tensors as a tree or a matrix;

`Length, Dimensions, TensorRank`, determining the tensor structure.

```
{Table[i1*i2,{i1,2},{i2,3}], Array[(#1*#2)&,{2,3}]}
tens={{{a,b,c},{d,e,f}},{{g,h,i},{j,k,l}}}
{TreeForm[tens], MatrixForm[tens], Length[tens],
Dimensions[tens],TensorRank[tens],TensorQ[tens]}
```

*Apply a function* to each element of an object:

`Map[f,expr]` or `f/@expr`, at the first level in `expr`;  
`Map[f,expr,levels]`, apply `f` to parts of `expr`;  
`Apply[f,expr]` or `f@@expr`, replace the head of the expression `expr` with the function `f`;  
`Thread[f[args]]`, `Thread[f[args],head]`, “threads” the function `f` over any objects with `head` that appear in the arguments `args` of the function `f`.

```
l1={{a,b},{c,d}}; l2={{e,f},{g,h}}; eq1=a==b;
{f1/@l1, Map[f2,l1,{2}], f3/@(x^3+x+2)}
(Apply[Plus,(Range[1,100])^2]//N)===
(Sum[i^2,{i,1,100}]//N)
{Thread[#^n&@eq1,Equal], Thread[eq1^n,Equal], eq1^n}
{l1==l2,Thread[l1==l2]}
```

**Problem:** A sequence of numbers  $\{x_i\}$  ( $i=0, \dots, n$ ) is defined by  $x_{i+1} = ax_i(1 + x_i)$  ( $0 < a < 10$ ), where  $a$  is a given parameter. Define a list of coordinates  $[i, x_i]$  such that  $a = 3$ ,  $x_0 = 0.1$ ,  $n = 100$ . Plot the graph of the sequence  $\{x_i\}$ .

```
n=100; l=Array[x,n,0]; a=3; l[[1]]=0.1;
Do[l[[i]]=a*l[[i-1]]*(1-l[[i-1]]),{i,2,n}]
l1=Table[{i,l[[i]]},{i,2,n}]
ListPlot[l1,PlotStyle->{PointSize[0.02],Hue[0.7]}];
```

**Problem:** Observe the function behavior  $y(x) = \cos(6(x - a \sin x))$ ,  $x \in [-\pi, \pi]$ ,  $a \in [\frac{1}{2}, \frac{3}{2}]$  (in more detail about animation, see Sect. 3.6).

```
<<Graphics`Animation`;
y[x_]:=Cos[6*(x-a*Sin[x])];
n=20;
Do[a=1/2+i/n;
g[i]=Plot[y[x],{x,-Pi,Pi},PlotRange->All,
DisplayFunction->Identity,PlotStyle->Hue[0.7]],{i,0,n}];
gr=Table[g[i],{i,0,n}];
ShowAnimation[gr,DisplayFunction->$DisplayFunction];
```

## Part II

### Mathematics: *Maple* and *Mathematica*

# Chapter 3

## Graphics

### 3.1 Simple Graphs

*Graphs of real values of expr or the functions  $f(x)$ ,  $f(x, y)$ ,  $x \in [x_1, x_2]$ ,  $(x, y) \in [x_1, x_2] \times [y_1, y_2]$ .*

*Maple:*

```
f:=expr;          plot(f, x1..x2);
f:=x->expr;      plot(f(x),x= x1..x2, ops);
f:=(x,y)->expr; plot3d(f(x,y),x=x1..x2,y=y1..y2,ops);
```

```
f1:=x->sin(x); f2:=sin(x); f3:=(x,y)->x^2-x-y^2-y-8;
plot(f1(x), x=-2*Pi..2*Pi); plot(f2, x=-2*Pi..2*Pi);
plot3d(f3(x,y),x=-6..6,y=-6..6,axes=boxed);
```

*Mathematica:*

```
Plot[f[x],{x, x1, x2}];
Plot3D[f[x,y], {x,x1,x2}, {y,y1,y2}];
```

```
Plot[Sin[x],{x,-2*Pi,2*Pi}];  f[x_,y_]:=x^2-x-y^2-y-8;
Plot3D[f[x,y],{x,-6,6},{y,-6,6},Boxed->True];
```

## 3.2 Various Options

In *Maple*, all the graphs can be drawn with various versions of `plot` and the package `plots`. The function `plot` has various forms (e.g., `logplot`, `odeplot`, `plot3d`) and various optional arguments which define the final figure (see `?plot[options]`, `?plot3d[options]`), e.g., light setting, legends, axis control, titles, gridlines, real-time rotation of 3D graphs, wide variety of coordinate systems, etc.

```

g1:=x->exp(-(x-3)^2*cos(Pi*x)^2);
plot(g1(x),x=0..6,tickmarks=[4,4],title='Graph of g(x)');
plot(sin(x)/x, x=-3*Pi..3*Pi, scaling = constrained);
plot(sin(x)/x, x=-3*Pi..3*Pi, style = point);
plot(tan(x), x=-2*Pi..2*Pi, y=-4..4, discont=true);
Points:=[[1,2],[2,3],[3,5],[4,7],[6,13],[7,17],[8,19]];
plot(Points, style = point); plot(Points, style = line);
g2:=(x,y)->x^2*sin(2*y)+y^2*sin(2*x);
plot3d(g2(x,y),x=0..Pi,y=0..Pi,grid=[20,20],style=patch);

```

In *Mathematica*, there are many options available for function graphics which can define the final picture (in more detail, see `Options[Plot]`, `Options[Plot3D]`), e.g., light modeling, legends, axis control, titles, gridlines, etc. The general rule for defining options is:

```

Plot[f[x], {x,x1,x2}, opName->value, ...]
Plot3D[f[x,y],{x,x1,x2},{y,y1,y2}, opName->value,...]

```

where `opName` is the option name.

The *description -Graphics-*, which follows the graphs, you can suppress by using `();`. Another possibility to suppress the graph output, is the option `DisplayFunction`.

*Formula for color graphs:* *Mathematica* makes it easy to compute the RGB formula for color graphs, using `Input → ColorSelector`. In addition, the package `Graphics`Colors`` contains a list of predefined colors, type `AllColors` to see all names, type one name, e.g., `Coral` to see RGB formula.

*Some useful options* for 2D graphs:

```
<<Graphics`Legend`;
<<Graphics`FilledPlot`;
f[x_]:=Exp[-(x-3)^2*Cos[Pi*x]^2];
Plot[f[x],{x,-Pi,2*Pi},PlotRange->All,PlotLabel->"f[x]"];
Plot[f[x],{x,-Pi,2*Pi},AxesLabel->{"x,sm","y,sec"}];
Plot[f[x],{x,0,2*Pi},AspectRatio->Automatic];
Plot[f[x],{x,-Pi,2*Pi},AxesOrigin->{3,0},
    PlotRange->{{2,4},{0,1}},Frame->True,Axes->False,
    GridLines->{{2,2.5,3,Pi,3.5,4},Automatic},
    FrameTicks->{Automatic,{0.2,0.8}}];
f1[x_]:=Sin[x]; f2[x_]:=Cos[x]; f3[x_]:=Sin[x]-Cos[x];
Plot[{f1[x],f2[x],f3[x]},{x,-Pi,Pi},
    PlotStyle->{GrayLevel[0.5], Dashing[{0.02,0.03}],
    Thickness[0.02]},PlotLegend->{f1[x],f2[x],f3[x]},
    Ticks->{{0,Pi/4,Pi/2,3*Pi/4,Pi},Automatic}];
Plot[{f1[x],f2[x]},{x,-Pi,Pi},PlotStyle->{Red,Blue}];
Plot[{f1[x],f2[x],f3[x]},{x,-Pi,Pi},PlotStyle->
    {RGBColor[0.501961,1,0],RGBColor[1,0.501961,1],
    RGBColor[1, 0.501961, 0]}];
```

*Options for 3D graphs:* most 2D graph options are valid for `Plot3D`. Here we present some useful special options for 3D graphs:

```
<<Graphics`Legend`;
<<Graphics`FilledPlot`;
f[x_,y_]:=Exp[-(x+y)];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},
    PlotPoints->{25,40}];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},
    Mesh->False, Boxed->False];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},Shading->False];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},BoxRatios->{1,2,1}];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},
    FaceGrids->All, Axes->False];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},
    HiddenSurface->False];
Plot3D[f[x,y],{x,-Pi,2*Pi},{y,-Pi,2*Pi},
    ViewPoint->{-1,2,1}];
```

In *Mathematica*, the `ViewPoint` parameters can also be determined interactively using `Input → 3DViewPointSelector`.

*The global options* for 2D and 3D graphs:

*Maple*:

the functions `setoptions`, `setoptions3d` (see `?plots`),

```
with(plots);
setoptions(axes=boxed, title='graph of f(x)');
setoptions3d(axes=normal, title='graph of g(x, y)');
```

*Mathematica*:

the function `SetOptions[symb, opt1a->val1, ...]`,

```
SetOptions[Plot, Frame->True, PlotLabel->"f[x]"];
Plot[{x^2, Sin[x^2]}, {x, -Pi, Pi}];
SetOptions[Plot3D, Boxed->False, PlotLabel->"f[x,y]"];
Plot3D[x^2+y^2, {x, -2, 2}, {y, -2, 2}];
```

### 3.3 Multiple Graphs

*A list or a set of graphs* in the same figure.

*Maple*:

```
L1:=[f1(x),...,fn(x)]: S1:={f1(x),...,fn(x)}:
L2:=[F1,...,Fn]: S2:={F1,...,Fn}:
plot(L1, x=a..b); plot(S1, x=a..b);
plot(L2, x= a..b); plot(S2, x=a..b);
```

where `L1, L2` and `S1, S2` are the lists and sets of functions and expressions, respectively.

```
f:=x->sin(x)/x; g:=x->cos(x)/x;
plot([f(x), g(x)], x=0..10*Pi, y=-1..2,
      linestyle=[SOLID, DOT], color=[red, blue]);
```

*Mathematica*:

```
Plot[{f1[x], f2[x], ..., fn[x]}, {x, x1, x2}];
```

```
Plot[{Cos[x], Sin[x]}, {x, -10*Pi, 10*Pi}];
```

*Merging* various saved graphic objects, an array of graphic objects.

*Maple*:

```
with(plots): L1:=[G1,...,Gn]: S1:={G1,...,Gn}:
L2:=[H1,...,Hn]: S2:={H1,...,Hn}:
display(L1, x=a..b); display(S1, x=a..b);
display3d(L2, x=a..b); display(S2, x=a..b);
G:=array(1..n):G[i]:=plot(fun_i,x=a..b):display(G);
```

where  $L_1, L_2$  and  $S_1, S_2$  are the lists and sets of saved 2D and 3D graphs, respectively.

```
with(plots);
f:=x->abs(sin(x)); g:=x->-cos(x);
G1:=plot(f(x), x=-Pi..Pi); G2:=plot(g(x), x=-Pi..Pi):
display({G1, G2}, title= "f(x) and g(x)");
G:= array(1..2);
G[1]:=plot(sin(x), x=-Pi..Pi):
G[2]:=plot(cos(x), x=0..Pi): display(G);
```

*Mathematica*:

```
Show[{g1,g2,...}]; GraphicsArray[{g11,g12,...},...];
<<Graphics`MultipleListPlot`;
MultipleListPlot[list1,...];
```

```
f1[x_] := Abs[Sin[x]]; f2[x_] := -Cos[x];
g1=Plot[f1[x],{x,-Pi,Pi},DisplayFunction->Identity];
g2=Plot[f2[x],{x,-2*Pi,2*Pi},DisplayFunction->Identity];
```

```

Show[{g1,g2},Frame->True,PlotLabel->"f1 and f2",
     DisplayFunction->$DisplayFunction,AspectRatio->1];
Show[GraphicsArray[{g1, g2}], Frame -> True,
     DisplayFunction->$DisplayFunction,AspectRatio->1];
ga2 = GraphicsArray[{{g1}, {g2}}];
Show[ga2,Frame->True,DisplayFunction->$DisplayFunction,
     Background->RGBColor[0.5,1.,1.]];
<<Graphics`MultipleListPlot`;
list1=Table[{i,i^3},{i,-5,5}];
list2=Table[{i,i^2},{i,-5,5}];
MultipleListPlot[list1, list2, PlotJoined -> True];
g3=Plot3D[2x^2-3y^2,{x,-1,1},{y,-1,1},
           DisplayFunction->Identity];
g4=Plot3D[3*x+y,{x,-1,1},{y,-1,1},
           DisplayFunction->Identity];
Show[g3,g4,BoxRatios->{1,1,2},
      DisplayFunction->$DisplayFunction];

```

**Problem:** Graph the stability diagram of the Mathieu differential equation  $x'' + [a - 2q \cos(2t)]x = 0$  in the  $(a, q)$ -plane (see Sect. 8.4).

*Maple:*

```

with(plots):
S1:=[MathieuA(i,q) $ i=0..5]; S2:=[MathieuB(i,q) $ i=1..6];
G1:=plot(S1,q=0..35,-20..45,color=red):
G2:=plot(S2,q=0..35,-20..45,color=blue):
display({G1,G2},thickness=3);

```

*Mathematica:*

```

s1=Table[MathieuCharacteristicA[i,q],{i,0,5}]
s2=Table[MathieuCharacteristicB[i,q],{i,1,6}]
g1=Plot[Evaluate[s1],{q,0,35},PlotStyle->{Red},
         DisplayFunction->Identity];
g2=Plot[Evaluate[s2],{q,0,35},PlotStyle->{Blue},
         DisplayFunction->Identity];
Show[{g1,g2},PlotRange->{{0,35},{-20,45}},Frame->True,
      PlotLabel->"Stability Regions",Axes->False,
      DisplayFunction->$DisplayFunction,AspectRatio->1];

```

### 3.4 Text in Graphs

*Drawing text strings* on 2D and 3D graphs.

*Maple:*

```
textplot([[x1,y1,Str1], ..., [xn,yn,Strn]],ops);
textplot3d([[x1,y1,z1,Str1], ..., [xn,yn,zn,Strn]],ops);
```

```
with(plots): f:=x->4*x^3+6*x^2-9*x+2;
G1 := plot([f(x), D(f)(x), (D@@2)(f)(x)], x=-3..3):
G2 := textplot([1.2, 100, "f(x) and their derivatives"],
               font=[HELVETICA,BOLD,13],color=plum):
display([G1,G2]);
P1:=plot3d(exp(-(x^2+y^2)),x=-6..6,y=-6..6,grid=[25,25]):
P2:=plot3d(-5-4*sin(sqrt(x^2+y^2)),x=-6..6,y=-6..6):
P3:=textplot3d([1,1,2,"a"],font=[SYMBOL,25],color=blue):
display3d([P1,P2,P3],orientation=[34,79]);
```

*Mathematica:*

the function `Graphics[Text["string",xtext,ytext]]`, the plot options `TextStyle`, `FormatType`,

```
f[x_]:=4*x^3+6*x^2-9*x+2;
g11=Plot[{f[x],-f'[x]},{x,-3,3},PlotStyle->{Red,Blue},
          TextStyle->{FontFamily->"Helvetica",FontSlant->
          "Italic",FontWeight->"Bold",FontSize->15},
          DisplayFunction->Identity];
g12=Graphics[Text["f[x] and -f'[x]", {0, 100}]];
Show[{g11, g12},Frame->True,Axes->False,PlotRange->All,
      DisplayFunction -> $DisplayFunction];
<<Graphics`Graphics3D`;
g31=Plot3D[Exp[-(x^2+y^2)],[x,-6,6],[y,-6,6],
           TextStyle->{FontFamily->"Symbol",FontSize->15},
           DisplayFunction->Identity];
g32=Plot3D[-5-4*Sin[Sqrt[x^2+y^2]],[x,-6,6],[y,-6,6],
           DisplayFunction -> Identity];
g33=Graphics3D[Text["a",{5,5,5}]];
Show[{g31,g32,g33},DisplayFunction->$DisplayFunction];
```

### 3.5 Special Graphs

*Coordinate lines* for 2D graphs.

*Maple:*

```
plot(f(x), x=x1..x2, gridlines=true);
with(plots): conformal(z, z=z1..z2, ops):
coordplot(coordsystem, [xrange, yrangle], ops);
```

*Mathematica:*

```
Plot[f[x], {x,x1,x2}, GridLines->{{xlist},{ylist}}];
```

**Problem:** Plot  $f(x) = x \sin(1/x)$  and  $g(x) = \frac{x^2 - x + 1}{x^2 + x - 1}$  together with the corresponding coordinate lines.

*Maple:*

```
with(plots): A:=4: f:=x->sin(1/x)*x;
plot(f(x),x=-Pi/2..Pi/2,color=blue,thickness=3,gridlines=true):
G1:=plot(f(x),x=-Pi/2..Pi/2,color=blue,thickness=3):
M1:=conformal(z,z=-A-I..A+I,grid=[20,10],color=grey):
display([G1,M1]): g:=x->(x^2-x+1)/(x^2+x-1); R:=-5..5;
G2 := plot(g(x),x=R,R,discont=true,thickness=3):
M2 := coordplot(cartesian, [R, R], view=[R, R],
grid=[10, 10], color=[grey, grey]):
display([G2, M2], axes=boxed , scaling=constrained,
xtickmarks=5, ytickmarks=5);
```

*Mathematica:*

```
f[x_]:=x*Sin[1/x]; g[x_]:=(x^2-x+1)/(x^2+x-1);
Plot[f[x],{x,-Pi/2,Pi/2},Frame->True,Axes->False,
PlotStyle->{Blue,Thickness[0.01]},
GridLines->{Automatic,Automatic}];
Plot[g[x],{x,-5,5},Frame->True,Axes->False,
PlotStyle->{Blue,Thickness[0.007]},
GridLines->{Automatic,Automatic}];
```

*Bounded regions* for 2D graphs.

*Maple:*

the function `inequal` (see `?plot[options]`),

```
plot(f(x), x=a..b, filled=true);
with(plots): inequal(ineqs, x=a..b, y=c..d, ops);
```

*Mathematica:*

```
FilledPlot[f1[x], ..., {x, x1, x2}];
```

```
<<Graphics`FilledPlot`;
FilledPlot[{Cos[x], Cos[2*x], Cos[3*x]}, {x, -Pi/2, Pi/2}];
FilledPlot[{Cos[x], Cos[2*x], Cos[3*x]}, {x, -Pi/2, Pi/2},
  Fills -> {GrayLevel[0.8], GrayLevel[0.6], Red}];
```

**Problem:** Plot the region bounded by  $f(x) = -(x + 1)^2 + 10$  and the axis  $x = 0$ .

*Maple:*

```
with(plots):
f:=x->-(x+1)^2+10: S:=[fsolve(f(x)=0,x)];
f1:=plot([f(x),0],x=-4.5..4.5,-2..10,thickness=3):
f2:=plot(f(x),x=S[1]..S[2],filled=true,color=grey):
display([f1,f2]);
```

*Mathematica:*

```
<<Graphics`FilledPlot`;
f[x_]:=- (x+1)^2+10; S=NSolve[f[x]==0,x]
FilledPlot[f[x],{x,S[[1,1,2]],S[[2,1,2]]}];
```

**Problem:** Plot the region that satisfies the inequality  $2x - 2y > 1$ .

*Maple:*

```
with(plots): Ineq:=x->2*x-2*y>1; A:=(color=blue);
B:=(color=grey); C:=(color=green,thickness=10);
inequal(Ineq(x),x=-2..2, y=-2..2,
optionsfeasible=A,optionsexcluded=B,optionsopen=C);
```

*Mathematica:*

```
<<IneqGraphics`; IneqPlot[2*x-2*y>1,{x,-2,2},{y,-2,2}];
```

where the package `IneqGraphics`` can be downloaded from the *Wolfram Information Center* according to the corresponding path in your computer, e.g., `Mathematica → 5.2 → AddOns → ExtraPackages`.

*Logarithmic plots* in the plane.

*Maple:*

```
with(plots): logplot(f,range,ops);
semilogplot(f,range,ops); loglogplot(f,range,ops);
```

```
with(plots); with(stats):
al:=stats[random, normald](20);
Points:=[seq([0.2*i,exp(0.1*i)+0.1*al[i]],i=1..20)];
G1:=logplot(Points, style=point, color=green):
G2:=logplot(x+sin(x),x=0.5..3,style=line,color=red):
display({G1, G2});
f:=x->x^5+exp(-x^5);
loglogplot(f(x),x=0.1..100,scaling=constrained);
```

*Mathematica:*

```
<<Graphics`Graphics`;
LogPlot[f,{x,x1,x2}];
LogLinearPlot[f,{x,x1,x2}]; LogLogPlot[f,{x,x1,x2}];
LogListPlot[{{x1,y1},{x2,y2},...}];
LogLinearListPlot[{{x1,y1},{x2,y2},...}];
LogLogListPlot[{{x1,y1},{x2,y2},...}];
```

```
<<Graphics`Graphics`;
<<Statistics`;
al=Table[Random[NormalDistribution[]],{20}]
points=Table[{0.2*i,Exp[0.1*i]+0.1*al[[i]]},{i,1,20}]
g1=LogListPlot[points,DisplayFunction->Identity,
  PlotStyle->{PointSize[0.02],Hue[0.7]}];
g2=LogPlot[x+Sin[x],{x,0.5,3},PlotStyle->Red,
  DisplayFunction->Identity];
Show[{g1,g2},DisplayFunction->$DisplayFunction];
f[x_]:=x^5+Exp[-x^5];
LogLogPlot[Evaluate[N[f[x],30]],{x,0.1,100}];
```

*Plots* of piecewise continuous functions.

*Maple*:

```
f := proc(x) if cond1 then expr1 else expr2 fi; end;
plot('f(x)',x=a..b); plot(f, a..b);
with(plots): f:=x->piecewise(cond1, expr1, expr2);
plot(f(x),x=a..b);
```

```
f:=proc(x) if x<0 then 0 elif x<1 then x else 1 fi; end;
with(plots); plot('f(x)',x=0..2); plot(f, 0..2);
g:=x->piecewise(x<0,0,x<1,x,1); plot(g(x),x=0..2);
```

*Mathematica*:

```
f1[x_]:=var1/;cond1; f[x_]:=var2/;cond2;
f2[x_]:=Piecewise[{{val1,cond1},{val2,cond2},...}];
f3[x_]:=UnitStep[x];Plot[{f1[x],f2[x],f3[x]},{x,a,b}];
```

```
f1[x_]:=-1/;x<0; f1[x_]:=x/;0<=x<=1; f1[x_]:=1/;x>1;
f2[x_]:=Piecewise[{{x,x<0},{-x,0<=x<=2},{1,x>2}}];
f3[x_]:=1/2*UnitStep[x]-UnitStep[x-1];
{Plot[f1[x],{x,-2,2},PlotStyle->Hue[0.5]],
 Plot[f2[x],{x,-3,3},PlotStyle->Hue[0.7]],
 Plot[f3[x],{x,-Pi,Pi},PlotStyle->Hue[0.9]]};
```

A density plot of the function  $f(x, y)$ ,  $(x, y) \in [x_1, x_2] \times [y_1, y_2]$ .

*Maple:*

```
with(plots): densityplot(f(x,y),x=x1..x2,y=y1..y2);
```

*Mathematica:*

```
DensityPlot[f[x,y],{x,x1,x2},{y,y1,y2}];  
ListDensityPlot[{{a11,...,a1n},...{an1,...,ann}}];
```

**Problem:** Construct the density plot of  $f(x, y) = xe^{-x^2-y^2}$ ,  $(x, y) \in [-2, 2] \times [-2, 2]$ , with color gradient and the corresponding legend.

*Maple:*

```
with(plots):  
A:=colorstyle=HUE,style=patchnogrid,numpoints=5000,axes=boxed;  
G1:=densityplot((x,y)->x*exp(-x^2-y^2),-2..2,-2..2,A):  
G2:=densityplot((x,y)->0.2*y, 3..3.5,-2..2,A):  
G3:=textplot([seq([3.8,-1.95+i/8*3.9,  
              sprintf("%.1f",-0.4+i/10)],i=0..8)]):  
display({G1, G2, G3}, scaling=constrained);
```

*Mathematica:*

```
f1[x_,y_]:=x*Exp[-x^2-y^2]; f2[x_,y_]:=0.2*y;  
g1=DensityPlot[Evaluate[f1[x, y],{x,-2,2},{y,-2,2},  
    Mesh->False,ColorFunction->Hue,PlotPoints->100,  
    DisplayFunction->Identity]];  
g2=DensityPlot[Evaluate[f2[x,y],{x,3,3.5},{y,-2,2},  
    Mesh->False,ColorFunction->Hue,PlotPoints->100,  
    DisplayFunction->Identity]];  
g3=Graphics[Table[Text[StyleForm[NumberForm[  
    N[-0.4+i/10],2],FontSize->9],  
    {2.7,-1.95+i/8*3.9}],{i,0,8}]];  
Show[g1,g2,g3,DisplayFunction->$DisplayFunction];  
list1=Table[Random[Real,{1,10}],{x,1,10},{y,1,10}];  
ListDensityPlot[list1,MeshRange->{{-8,8},{-8,8}}];
```

*Bar graphs:* different types of 2D and 3D bar graphs.

*Maple:*

```
with(stats):
with(stats[statplots]):histogram(list1,list2,...,ops);
with(Statistics):
PieChart(list,ops);      PieChart['interactive'](list);
```

*Mathematica:*

```
<<Graphics`Graphics`;
<<Graphics`Graphics3D`;
BarChart[list1,list2,...,BarOrientation->Horizontal,
  BarEdges->True,BarValues->True,BarLabels->"list",
  BarSpacing->value,BarStyle->style]
StackedBarChart[list1,list2,...]
PercentiledBarChart[list1,list2,...]
GeneralizedBarChart[list1,list2,...]
PieChart[{x1,x2,...},PieLabels->list,PieExploded->All,
  PieStyle->style,PieLineStyle->style]
BarChart3D[{{z11,...},{z21,...},...},
  BarLabels->"list",XSpacing->value,YSpacing->value,
  SolidBarStyle->style]
```

**Problem:** Compare different bar graphs.

*Maple:*

```
with(stats): with(stats[statplots]):
list1 := [random[normald](200)]:
histogram(list1); histogram(list1,area=count);
histogram(list1,color=blue);
with(Statistics): list2:=[i^2 $ i=1..5]:
PieChart(list2,sector=0..360,
  color=[blue,red,green,yellow,white]);
```

*Mathematica:*

```
<<Graphics`Graphics`;
<<Graphics`Graphics3D`;
list1=Table[Random[Integer,{1,5}],{i,1,5}];
list2=Table[i^2,{i,1,5}]; list3={"a1","a2","a3","a4","a5"};
g1=BarChart[list1,BarStyle->{GrayLevel[0.5]},
    BarEdges->True,BarOrientation->Horizontal,
    DisplayFunction->Identity];
g2=BarChart[{list1,list2},BarStyle->{GrayLevel[0.5],
    GrayLevel[0]},DisplayFunction->Identity];
g3=StackedBarChart[{list1,list2},DisplayFunction->Identity,
    BarStyle->{GrayLevel[0.5],GrayLevel[0]}];
g4=PercentileBarChart[{list1,list2},BarLabels->list3,
    BarStyle->{GrayLevel[0.5],GrayLevel[0]},
    BarSpacing->0.2,DisplayFunction->Identity];
g5=PieChart[{list1},PieLabels->list3,PieExploded->All,
    PieStyle->{GrayLevel[0.1],GrayLevel[0.6],
    GrayLevel[0.7],GrayLevel[0.8],GrayLevel[0.9]},
    DisplayFunction->Identity];
g6=PieChart[{list1},PieLabels->list3,PieExploded->{1},
    PieStyle->{GrayLevel[0.1],GrayLevel[0.6],
    GrayLevel[0.7],GrayLevel[0.8],GrayLevel[0.9]},
    DisplayFunction->Identity];
g=GraphicsArray[{{g1,g2},{g3,g4},{g5,g6}}];
Show[g,DisplayFunction->$DisplayFunction];
list4={{1,2,3},{6,5,4},{9,10,11}};
BarChart3D[list4,AxesLabel->{"x","y","z"},
    BarLabels->{"1","2","3"},XSpacing->0.3,
    YSpacing->0.3,AmbientLight->Hue[Random[]]];
```

## 3.6 Animations

*2D and 3D animations of functions.*

*Maple:*

```
with(plots): animatecurve(f(x), x=a..b, ops);
animate(f(x,t), x=a..b, t=t1..t2, ops);
animate3d(f(x,y,t), x=a..b, y=c..d, t=t1..t2, ops);
display([G1,G2,...,GN], insequence=true);
display3d([G1,G2,...,GN], insequence=true);
```

*Mathematica:*

*2D and 3D animations* of plot sequences can be produced using the option `Animate Selected Graphics` of the `Cell` menu. After the construction of a plot sequence, select the second cell bracket (double-click) and select the top image (double-click) or go to `AnimateSelectedGraphics`.

For the construction of plot sequences the following functions can be used:

`Animate`, `ShowAnimation`, `MoviePlot`, `MovieParametricPlot`, `Do`, etc. (see `?Movie*`)

```
<<Graphics`Animation`;
Do[g,{t,t1,t2,tStep}];
Animate[g,{t,t1,t2,tStep}]; ShowAnimation[g1,...,gn];
MovieParametricPlot[{f[x,t],g[x,t]},{x,x1,x2},{t,t1,t2}];
MoviePlot[f[x,t],{x,x1,x2},{t,t1,t2}];
```

```
<<Graphics`Animation`;
f[x_,t_]:=Sin[x+t]+Sin[x-2*t];
g1=Animate[Plot[Sin[-(x-t)^2],{x,-2,2}],{t,0.2,0.5}];
g2=Animate[Plot[f[x,t],{x,-10,10},
    PlotRange->{-10,10}],{t,0,30,1}];
g3=Animate[Plot3D[Sin[x-y-t],{x,0,Pi},{y,0,Pi}],{t,0,2*Pi}];
ShowAnimation[g1,g2,g3];
MoviePlot[x*Sin[x*t],{x,-2*Pi,0},{t,0.2,0.5},
    PlotRange->{-6,6}];
Do[Plot[Sin[x*i],{x,0,Pi},PlotRange->{-1,1}],{i,1,10}];
```

**Problem:** Show that the solutions  $u_1(x, t) = \cos(x - 2t)$  and  $u_2(x, t) = \cos(x + 2t)$ , of the wave equation are traveling waves.

*Maple:*

```
with(plots): N:=200; A:=array(1..2):
u1:=(x,t)->cos(x-2*t); u2:=(x,t)->cos(x+2*t);
setoptions(thickness=3,scaling=constrained,axes=boxed);
A[1]:=animate(u1(x,t),x=0..4*Pi,t=1..10,frames=N):
A[2]:=animate(u2(x,t),x=Pi/2..9*Pi/2,t=1..10,
    color=green,frames=N):
display(A);
```

*Mathematica:*

```
<<Graphics`Animation`;
u1[x_]:=Cos[x-2*t]; u2[x_]:=Cos[x+2*t];
SetOptions[Plot,Frame->True,PlotStyle->Thickness[0.02],
           AspectRatio->1];
g=Table[{Plot[u1[x],{x,0,4*Pi},PlotStyle->Red,
DisplayFunction->Identity], Plot[u2[x],{x,Pi/2,9*Pi/2},
PlotStyle->Blue,DisplayFunction->Identity]}, {t,1,5}];
ShowAnimation[g,DisplayFunction->$DisplayFunction];
```

**Problem:** Observe the Lissajous curves in polar coordinates.

*Maple:*

```
with(plots):
animatecurve([sin(7*x),cos(11*x),x=0..2*Pi],coords=polar,
numpoints=300,frames=300,color=blue,thickness=3);
```

*Mathematica:*

```
<<Graphics`Animation`;
MovieParametricPlot[{Sin[n*t],Cos[(n+2)*t]},{t,0,2*Pi},
{n,1,5,2},PlotStyle->Blue,Frame->True,FrameTicks->False];
```

**Problem:** Animations of two sequences of points in 2D.

*Maple:*

```
with(plots): n:=100: G:=[];
L1:=[seq([cos(j*Pi/n),sin(j*Pi/n)],j=0..n)]:
L2:=[seq([-cos(j*Pi/n),-sin(j*Pi/n)],j=0..n)]:
for i from 1 to n do
G:=[op(G),plot([L1[1..i],L2[1..i]], x=-1..1,y=-1..1,
symbol=circle,style=point,color=[blue,red])]: od:
display(G,insequence=true);
```

*Mathematica:*

```
<<Graphics`Animation`;
<<Graphics`MultipleListPlot`;
n=100; g={};
l1=Table[{Cos[j*Pi/n],Sin[j*Pi/n]},{j,0,n}]//N
l2=Table[{-Cos[j*Pi/n],-Sin[j*Pi/n]},{j,0,n}]//N
Do[g=Append[g,Evaluate[MultipleListPlot[
    Take[l1,{1,i}],Take[l2,{1,i}],SymbolStyle->{Blue,Red},
    SymbolShape->{PlotSymbol[Triangle,5],
    PlotSymbol[Triangle,5,Filled->False]},PlotRange->{-1,1},
    DisplayFunction->Identity]]],{i,1,n}];
ShowAnimation[g,DisplayFunction->$DisplayFunction];
```

**Problem:** Animations of two sequences of points in 3D.

*Maple:*

```
with(plots): n:=250: G:=[]: k:=20:
L1:=[seq([cos(k*j*Pi/n),j,sin(k*j*Pi/n)],j=0..n)]:
L2:=[seq([-cos(k*j*Pi/n),-j,-sin(k*j*Pi/n)],j=0..n)]:
for i from 1 to n do
    G:=[op(G),spacecurve({L1[1..i],L2[1..i]},style=line,
        axes=none,thickness=3,shading=zhue)]: od:
display3d(G,insequence=true);
```

*Mathematica:*

```
<<Graphics`Graphics3D`;
<<Graphics`Animation`;
n=500; g={}; k=20;
l1=Table[{Cos[k*j*Pi/n],j/n,Sin[k*j*Pi/n]},
    {j,0,n}]//N;
l2=Table[{-Cos[k*j*Pi/n],-j/n,-Sin[k*j*Pi/n]},
    {j,0,n}]//N;
Do[g=Append[g,Evaluate[{ScatterPlot3D[Take[l1,{1,i}],
    PlotJoined->True,PlotStyle->{Blue,PointSize[0.02]},
    PlotRange->{{-1,1},{-1,1},{-1,1}},
    DisplayFunction->Identity],
    ScatterPlot3D[Take[l2,{1,i}],PlotJoined->True,
    PlotStyle->{Red,PointSize[0.02]},
    PlotRange->{{-1,1},{-1,1},{-1,1}},
    DisplayFunction->Identity]}]],{i,1,n}];
ShowAnimation[g, DisplayFunction -> $DisplayFunction];
```

**Problem:** Animations of two structures in 3D.

*Maple:*

```
with(plots): n:=40: G:=NULL:
f1:=(x,y)->(x^2-y^2)/9-9: f2:=(x,y)->(x^2-y^2)/9:
G1:=plot3d(f1(x,y),x=-9..9,y=-9..9,style=patchnogrid):
G2:=plot3d(f2(x,y),x=-9..9,y=-9..9,style=patchnogrid):
for i from 1 to n do
  G:=G,display3d([G1,G2],orientation=[i*180/n,50]): od:
display3d([G],scaling=constrained,insequence=true);
```

*Mathematica:*

```
<<Graphics`Graphics3D`;
<<Graphics`Animation`;
n=40; g={};
f1[x_,y_]:=(x^2-y^2)/9-9; f2[x_,y_]:=(x^2-y^2)/9;
g1=Plot3D[f1[x,y],{x,-9,9},{y,-9,9},
  DisplayFunction->Identity,Mesh->False];
g2=Plot3D[f2[x,y],{x,-9,9},{y,-9,9},
  DisplayFunction->Identity,Mesh->False];
Do[g=Append[g,Show[{g1,g2},ViewPoint->{1.2+i/10,1.2,1.2},
  Boxed->False,Axes->False,PlotRange->
  {{-10,10},{-10,10},{-10,10}}]],{i,1,n}];
ShowAnimation[g,DisplayFunction->$DisplayFunction];
```

**Problem:** Observe the motion of a point rolling along a curve.

*Maple:*

```
with(plots):
G:=plot(cos(x),x=-Pi..Pi,axes=boxed,thickness=2,color=blue):
animate(pointplot,[[[t,cos(t)]],symbol=circle,symbolsize=20,
  color=red],t=-Pi..Pi, frames=100, background=G);
```

*Mathematica:*

```
<<Graphics`Animation`;
g=Plot[Cos[x],{x,-Pi,Pi},Frame->True,
  PlotStyle->{Blue,Thickness[0.02]},DisplayFunction->Identity];
Animate[Show[g,Graphics[Disk[{x,Cos[x]},0.2]],
  PlotRange->{-1.5,1.5},DisplayFunction->$DisplayFunction],
 {x,-Pi,Pi}];
```

# Chapter 4

## Algebra

### 4.1 Polynomials

*Maple:*

*Definition* of polynomials: univariate, multivariate polynomials, and polynomials over a number ring and a field can be defined (see `?polynomials`).

*Manipulation* of polynomials:

`coeff, coeffs, lcoeff, tcoeff`, extract polynomial coefficients;  
`degree, ldegree`, determine the degree and the lowest degree of a polynomial;  
`collect`, group the coefficients of like power terms together;  
`divide`, perform exact polynomial division;  
`roots`, find exact roots of an univariate polynomial over an algebraic number field;  
`sort`, a polynomial sorting operation;  
`type[polynom]`, test for polynomials, etc.

```
coeff(p,x,n);   coeffs(p,x);      lcoeff(p,x);
tcoeff(p,x);    degree(p,x);     ldegree(p,x);
collect(p,x);   divide(p,q);     roots(p,x);
sort(p,x);      type(p, polynom(domaincoef, x));
```

*Mathematica:*

*Definition* of polynomials: polynomials are represented beginning with the constant term.

*Manipulations* of polynomials:

**PolynomialQ**, test for polynomials;  
**Coefficient**, **CoefficientList**, extract polynomial coefficients;  
**Variables**, determine a list of all independent variables;  
**Roots**, **NRoots**, find exact and numerical solutions of polynomial equations;  
**Exponent**, determine the degree of a polynomial;  
**Collect**, group the coefficients of like terms together;  
**PolynomialQuotient**, **PolynomialRemainder**, perform exact polynomial division;  
**PolynomialGCD**, find the greatest common divisor of polynomials, etc.

```

PolynomialQ[expr,x];          Coefficient[p,x];
Coefficient[p,x,n];          CoefficientList[p,x];
Variables[p]; Roots[eq,x];   NRoots[eq,x]; Exponent[p,x];
Collect[p,x];                PolynomialQuotient[p,q,x];
PolynomialRemainder[p,q,x];  PolynomialGCD[p1,p2,...];
  
```

**Problem:** Define the univariate polynomial  $y = a_n x^n + \dots + a_1 x + a_0$ ,  $n = 10$ .

*Maple:*

```

n := 10;    y := add(a || i*x^i, i=0..n);
y := add(cat(a, i)*x^i, i=0..n);
y := a0 + add(cat(a, i)*x^i, i=1..n);
y := a0 + sum('cat(a, i)*x^i', 'i'=1..n);
y := convert(['a || i*x^i' $ 'i'=0..n], '+');
a := array(0..n); S := a[0];
for i from 1 to n do S:=S + a[i]*x^i; od: y:=S;
  
```

*Mathematica:*

```
c=Table[a[i],{i,0,10}]; u=Table[x^i,{i,0,10}]; c.u
f[x_,n_]:=Table[a[i],{i,0,n}].x^Range[0, n]; p=f[x,10]
{Coefficient[p,x,5], CoefficientList[p,x], Exponent[p,x]}
```

**Problem:** Find all the solutions of  $x^3 - 5x^2 - 2x + 10 = 0$  which are less than 1.

*Maple:*

```
sols:=[solve(x^3-5*x^2-2*x+10=0,x)];
numsol:=fsolve(x^3-5*x^2-2*x+10=0,x);
selectSol:= proc(list) local i, l; l:=NULL;
for i from 1 to nops(list) do
  if evalf(list[i])<1 then l:=l,list[i]; fi; od;
RETURN(l); end;
l1:= selectSol(sols); l2:= selectSol(numsol);
```

*Mathematica:*

```
sols=Roots[x^3-5*x^2-2*x+10==0,x]; sols&&x<1//Simplify
numsol=NRoots[x^3-5*x^2-2*x+10==0,x];
numsol&&x<1//Simplify
```

## 4.2 Simplification

*Maple:*

*Algebraic simplification*, operations for regrouping terms:

- `factor`, factorize over an algebraic number field;
- `expand`, distribute products over sums;
- `combine`, combine terms into a single term;
- `evala`, `eval`, evaluate in an algebraic number field;
- `simplify`, apply simplification rules to expressions;
- `normal`, find factored normal form;
- `numer`, `denom`, extract the numerator and denominator of an expression;

**convert**, change the form of an expression;  
**map**, manipulations of large expressions;  
**match**, pattern matching;  
**frontend**, extension of the computation domain for many functions, etc.

```
factor(expr); expand(expr);          combine(expr,ops);
evala(expr);  simplify(expr,ops);    normal(expr);
numer(expr);  denom(expr);         convert(expr,form,var);
map(function,expr);      match(expr=pattern,var,'s');
frontend(expr,x);
```

```
f := (x^3+2*x^2-x-2)/(x^3+x^2-4*x-4);
N1 := factor(numer(f)); N2 := subs(x=2, N1);
D1 := factor(denom(f)); D2 := subs(x=3, D1);
F1 := simplify(f);           F2 := subs(x=4, F1);
F3 := subs(x=-3, F1);      convert(f, parfrac, x);
F:=(x+y+1)^3*(x+3*y+3)^2;
B:=collect(F,x); C:=map(factor,B);
assume(x>0, y>0, t>0);
expand([ln(x*y), ln(x^y), (x*y)^z]);
combine([exp(t)*exp(p), t^p*t^q, ln(t)+ln(p)]);
```

*Mathematica:*

*Algebraic simplification*, operations for regrouping terms:

**Factor**, **FactorTerms**, factorize over an algebraic number field or over a list of algebraic numbers;  
**Short**, **Map** or **/@**, **Apply** or **@@**, **Thread**, manipulations of large expressions;  
**Expand**, **PowerExpand**, distribute products over sums;  
**Apart**, convert **expr** into a sum of partial fractions;  
**Together**, combine terms in a sum over a common denominator and cancel factors in the result;  
**Evaluate**, evaluation of an expression;

- Hold, maintain `expr` in unevaluated form;
- Cancel, common-factor cancellation in the numerator and denominator of `expr`;
- Numerator, Denominator, extract the numerator and denominator of `expr`;
- Simplify, FullSimplify, apply simplification rules to expressions or doing simplifications using assumptions, `assum`, etc.

Factor[expr];	FactorTerms[expr,x];	Short[expr];
Map[f,expr];	Apply[f,expr];	Thread[f[args]];
f /@ expr;	f @@ expr;	Thread[f[args],head];
Expand[expr];	PowerExpand[expr];	Apart[expr];
Together[expr];	Evaluate[expr];	Hold[expr];
Cancel[expr];	Numerator[expr];	Denominator[expr];
Simplify[expr];	FullSimplify[expr];	Simplify[expr,{assum}];

```

expr1=(x^3+2*x^2-x-2)/(x^8-21*x^4-100);
expr2=(x+y)^2*(3*x-y)^3; expr3=(x+y+1)^3/(x+3*y+3)^2;
Factor[Numerator[expr1]] /. x->2
Factor[Denominator[expr1], GaussianIntegers->True]
Factor[Denominator[expr1], Extension->{I, Sqrt[5]}]
ExpandNumerator[expr3]; ExpandDenominator[expr3]
expr4 = ExpandAll[expr3]; {Expand[expr3], Collect[expr2, x]}
{Log[x*y], Log[x^y], (x*y)^z} // PowerExpand
{Exp[t]*Exp[p], t^p*t^q} // Together
Cancel[expr1*(x + 1 + y)/expr2]
{Apart[expr1], FullSimplify[expr4], Map[Factor, expr4]}
Expand[expr4^10]//Short
{Simplify[(x^m)^n], Simplify[(x^m)^n, {m \ [Element] Integers,
n \ [Element] Integers}]}
Simplify[Log[x-y]+Log[x+y], {x>y}]

```

Note that the function `Element[x,domain]` specifies that  $x \in \text{domain}$ . The element operator ( $\in$ ) can be entered with the sequence Esc elem Esc. Assumptions can be employed with some other functions, for example, `FullSimplify`, `FunctionExpand`, `Refine`, `Limit`, `Integrate`.

**Problem:** Perform the inversion of the series of  $\tan(x)$  around of  $x = 0$ .

*Maple:*

```
Order:=20; Ec1:=y=series(tan(x),x=0,Order);
x:=solve(Ec1,x); x:=convert(x,polynom);
```

*Mathematica:*

```
f[t_]:=Normal[InverseSeries[
  Series[Tan[x],{x,0,20}]]/.x->t]; f[z]
```

*Trigonometric simplification* and manipulation.

*Maple:*

```
simplify(expr,trig); combine(expr,trig);
testeq(expr1==expr2);
```

*Mathematica:*

Setting the option `Trig->True` within a function, *Mathematica* will use standard trigonometric and hyperbolic identities to manipulate and simplify the expression, the specialized functions, `TrigExpand`, `TrigReduce`, `TrigFactor`, `TrigToExp`, `ExpToTrig`,

<code>Cancel[expr,Trig-&gt;True]</code>	<code>Together[expr,Trig-&gt;True]</code>
<code>TrigExpand[expr]</code>	<code>TrigReduce[expr]</code>
<code>TrigFactor[expr]</code>	<code>TrigToExp[expr]</code>
	<code>ExpToTrig[expr]</code>

**Problem:** Simplify the trigonometric expressions  $A = \frac{\sec^4 x - 1}{\tan^2 x}$  and  $B = \sin^3 x + \cos^3 x$ . Transform the expression for  $B$  to a sum of trigonometric terms with arguments  $x, 3x$ .

*Maple:*

```
A:=(sec(x)^4-1)/tan(x)^2; B:=sin(x)^3+cos(x)^3;
expand(simplify(A,trig));simplify(B,trig);combine(B,trig);
```

*Mathematica:*

```
A=(Sec[x]^4-1)/Tan[x]^2; B=Sin[x]^3+Cos[x]^3;
{Factor[{A,B},Trig->True], TrigReduce[{A,B}],
 Together[A,Trig->True]}
```

**Problem:** Show that the trigonometric expressions  $\sin 2x = 2 \sin x \cos x$  and  $\cos(x + y) = \cos x \cos y - \sin x \sin y$  are identical.

*Maple:*

```
testeq(sin(2*x)=2*sin(x)*cos(x));
testeq(cos(x+y)=cos(x)*cos(y)-sin(x)*sin(y));
```

*Mathematica:*

```
(Cos[x+y]//TrigExpand) === Cos[x]*Cos[y]-Sin[x]*Sin[y]
(Sin[2*x]//TrigExpand) === 2*Sin[x]*Cos[x]
```

## 4.3 Linear Algebra

*Linear algebra* is the algebra of linear spaces and linear transformations between these spaces. Linear algebra is one of the fundamental areas of Mathematics and one of the most common applied fields.

*Maple functions* for working with linear algebra are contained in the largest packages `linalg`, `LinearAlgebra`, `VectorCalculus`. We discuss the most important functions.

In *Mathematica*, vectors, matrices, tensors are represented as lists (see Subsect. 2.3.6). A variety of functions for working problems in linear algebra are contained in the package

`LinearAlgebra`MatrixManipulation``.

*Vector representations* and components.

*Maple:*

```
with(linalg): with(LinearAlgebra):
V1:= <1,2,3>; V2:=[1,2,3]; V3:=[4,5,6];
U:=Vector(3, i->u||i); W:=vector(3);
Vector([1,2,3]); Vector[row]([1,2,3]);
print(V1); op(U); print(U); V2[2]; U[1]:=w1; U[2]:=w2;
```

*Mathematica*:

**Table**, **Array**, vector representations;  
**ColumnForm**, **MatrixForm**, displaying the elements of list in the vector or matrix form;  
 $[[\cdot]]$ , **Take**, extracting a particular vector element or column;  
**VectorQ**, determine whether or not **expr** has the structure of a vector.

```
v1=Table[expr,{i,i1,in}];      v2=Array[f,n];
ColumnForm[v1];                MatrixForm[v1];
v1[[i]];                      Take[v1,{i1,i2}];
```

```
f1[i_]:=(-1)*i^3;
{v1={1,2,3,4,5}, v2=Table[f1[i],{i,5}], v3=Array[f1,5]}
{v1//ColumnForm, MatrixForm[v3], v3//MatrixForm,
 VectorQ[v2],v1[[3]], Take[v2,{1,4}]}
```

*Matrix representations* and components.

*Maple*:

```
with(linalg): with(LinearAlgebra):
M1:=<<1,2,9>|<2,3,9>|<3,4,9>|<5,6,9>>;
M2 := matrix(3,3); M3:=array(1..3,1..3,[]);
M4:=Matrix([[1,2,3],[4,5,6],[7,8,9]]);
M5:=matrix(3,3, (i,j)->a||i||j);
op(M4); M4[2, 2]:=-1; print(M4); M1[2, 2];
A:=matrix(2,2); entermatrix(A); print(A);
```

Note that for interactive matrix representation, **entermatrix**, the value entered must be terminated with a symbol **(;)**.

*Mathematica*:

**Table**, **Array**, matrix representations;  
**Input** → **CreateTable**/**Matrix**/**Palette**, introducing matrices;  
**MatrixForm**, displaying the elements of list in matrix form;

`MatrixQ`, determine whether or not `expr` has the structure of a matrix;

`[[]]`, `Take`, extracting a particular matrix element or a row, or a column, a submatrix.

```
m1=Table[expr,{i,i1,im},{j,j1,jn}];  
m2=Array[f,{m,n}]; m2//MatrixForm; MatrixQ[m2];  
m2[[i,j]]; m2[[i]]; m2[[All,j]];  
m1[[{i1,j1},{i2,j2}]]; Take[m1,{i1,j1},{i2,j2}];
```

```
f2[i_,j_]:=(-1)*(i*j)^3;  
{m1=Table[i*j,{i,3},{j,5}], m2=Array[f2,{3,5}]}  
{MatrixForm[m1], m2//MatrixForm, MatrixQ[m2]}  
m2[[2]]  
Map[MatrixForm,{m2[[All,2]],m2[[{1,2},{3,5}]],  
m2[[{1,2},All]],Take[m2,{1,3},{2,4}]}]
```

*Vector* algebra.

*Maple*:

```
with(linalg): with(LinearAlgebra):  
V1:= <1|2|3>; V2:=<4|5|6>; W:=vector(3);  
V3:= <1,2,3>; V4:=<4,5,6>; evalm(V3+V4);  
VectorAdd(V3,V4); Add(V3,V4); Add(V1,V2);  
2*V1; evalm(2*V1); Add(V1,V2,2,0);  
matadd(V3,V4,c1,c2); evalm(c1*V3+c2*V4);  
Add(V3,V4,c1,c2); matadd(V1,V2,1,-1);  
evalm(V1-V2); Add(V1,-V2);  
V:=2*V1+V2; V=evalm(V); W_m:=convert(W,matrix);  
W_v:=convert(W_m,vector); equal(W_v, W);  
print(W_m); type(W_m, matrix);
```

*Mathematica*:

```
Map[ColumnForm,{v1={1.,2.,3.}, v2={4.,5.,6.}}]  
Map[MatrixForm,  
{v1+v2, v1-v2, 2*v1, c1*v1+c2*v2, v1^3,  
v1.v2, Dot[v1,v2], Cross[v1,v2],  
Inner[Times,v1,v2,Plus], Outer[Times,v1,v2]}]
```

*Matrix* algebra.

*Maple*:

```
with(linalg): with(LinearAlgebra):
M1:=<<1,2,9>|<2,3,9>|<3,4,9>|<5,6,9>>;
M2 := matrix(3,3); W:=vector(3);
M3:=Matrix([[1,2,1],[4,5,4],[7,8,9]]);
M4:=Matrix(3,3, (i,j)->a||i||j);
eval(2*M2); evalm(M3*(-1)); MatrixAdd(M3,M4); Add(M3,M4);
multiply(M4,W); evalm(M4&*W); Multiply(M3,M4);
MatrixVectorMultiply(M1, <0,K,0,K>);
MatrixMatrixMultiply(M3,M4); MatrixScalarMultiply(M1,9);
factor(evalm(M3^5)); evalm(M3^(-3)); evalm(M3^0);
map(ln, evalm(M1));
```

*Mathematica*:

```
Map[MatrixForm,{v1={1,2,3},
  m1=Table[Random[Real,{0,1}],{i,1,3},{j,1,3}],
  m2=Table[If[i<=j,0,1],{i,1,3},{j,1,3}]]]
Map[MatrixForm,{m1+m2, m1-m2, 2*m1, c1*m1+c2*m2,
  MatrixPower[m1,-3], Log[m1], MatrixExp[m2],
  Inner[f,m1,m2,g], Outer[Times,m1,m2}}]//TableForm
```

*Matrix* manipulation.

*Maple*:

```
with(linalg): with(LinearAlgebra):
M1:=<<1,2,9>|<2,3,9>|<3,4,9>|<5,6,9>>;
M2 := matrix(3,3); W:=vector(3);
M3:=Matrix([[1,2,1],[4,5,4],[7,8,9]]);
M4:=Matrix(3,3, (i,j)->a||i||j);
eval(2*M2); evalm(M3*(-1)); MatrixAdd(M3,M4); Add(M3,M4);
multiply(M4,W); evalm(M4&*W); Multiply(M3,M4);
MatrixVectorMultiply(M1, <0,K,0,K>);
MatrixMatrixMultiply(M3,M4); MatrixScalarMultiply(M1,9);
ScalarMultiply(M1,C);
factor(evalm(M3^5)); evalm(M3^(-3)); evalm(M3^0);
map(ln, evalm(M1));
```

```
DeleteRow(M1,1..2);DeleteColumn(M1,[1,2]);DeleteRow(M1,1);
RowOperation(M1,3,2); ColumnOperation(M1,[1,3]);
RowOperation(M1,[1,-1],9); SubMatrix(M1,[-1,1],[1..2, 2]);
```

*Mathematica:*

```
<<LinearAlgebra`MatrixManipulation`;
m1=Table[Random[Integer,{0,10}],{i,1,4},{j,1,4}]
m2=Table[If[i<j,Sin[Pi*(i+j)],1],{i,1,4},{j,1,4}]
Map[MatrixForm,{m1,m2}]
Map[MatrixForm,{AppendRows[m1,m2], AppendColumns[m1,m2],
  TakeRows[m1,{1,3}], TakeColumns[m1,{2,4}],
  TakeMatrix[m1,{1,1},{3,3}], SubMatrix[m1,{1,2},{2,2}]}]
```

*Vector and matrix visualization.*

*Maple:*

```
with(plots):
V1:=[1,2,3]; V2:=[4,5,6]; v1:=[1,2]; v2:=[3,4];
M :=Matrix([[1,2,3],[4,5,6],[7,8,9]]);
pointplot3d({V1+V2,V2-2*V1}, color=blue,symbol=circle,
  symbolsize=20,axes=boxed,
  view=[-9..9,-9..9,-9..9]);
pointplot({evalm(v1-2*v2), evalm(v1+v2)},
  view=[-9..9,-9..9],color=blue,symbol=circle,
  symbolsize=20,axes=boxed);
matrixplot(M, heights=histogram,gap=0.3,
  style=patchnogrid);
```

*Mathematica:*

```
<<Graphics`Graphics3D`; <<Graphics`FilledPlot`;
<<Graphics`Graphics`;
<<LinearAlgebra`MatrixManipulation`;
v1={1,2,3}; v2={4,5,6}; v3={1,2}; v4={3,4};
m={{1,2,3},{4,5,6},{7,8,9}};
ScatterPlot3D[{v1+v2, v2-2*v1},
  PlotStyle->{Blue,PointSize[0.05]},Boxed->True,
  PlotRange->{{{-9,9},{{-9,9},{{-9,9}}}}];
```

```

ListPlot[{v3+v4, v3-2*v4},
  PlotStyle->{Blue,PointSize[0.05]},Frame->True,
  PlotRange->{{{-9,9},{-9,9}}];
ListPlot3D[m, BoxRatios->{1,1,1}];
FilledListPlot[{v3,v4}, Fills->Hue[0.7]];
Show[Graphics3D[{PointSize[0.03],Hue[0.9],Point[v1],
  Point[v2],Point[(v2-v1)/2]}],BoxRatios->{1,1,1}];
Show[Graphics[{PointSize[0.03],Hue[0.9],Point[v3],
  Point[v4],Point[v3-v4]}],Frame->True];
MatrixPlot[Table[Random[Real],{10},{10}],
  ColorFunction->(RGBColor[1, #, 1]&)];

```

*Vector spaces*, basis for the row or column space of a matrix, a basis for the nullspace (kernel) of a matrix, dimension.

*Maple*:

```

with(linalg): with(LinearAlgebra):
S1:= {<1,2,3>, <4,5,6>, <7,8,9>};
S2:= [<1,2,3>, <4,5,6>, <7,8,9>];
S3:= {vector([1,1,3]),vector([2,4,6]),vector([3,5,9])};
v:=<1,1,1>; A:=<<1,3,1>|<1,3,1>|<2,4,1>>;
B_S1:= Basis(S1); B_S2:=Basis(S2); B_S3:=basis(S3);
RowSpace(A); ColumnSpace(A); NullSpace(A);
for i from 1 to 3 do
  Dimension_S||i := nops(B_S||i); od;
SumBasis([S2,v]); IntersectionBasis([S2,v]);
Dimension(v); Dimension(A);
RowDimension(A); ColumnDimension(A);

```

*Mathematica*:

```

<<LinearAlgebra`MatrixManipulation`;
{m1={{1,4,7},{2,5,8},{3,6,9}}},
 m2={{1,1,2},{3,3,4},{1,1,1}}}
{rowSpace=DeleteCases[RowReduce[m2],_?(Union[#]=={0}&)],
 columnSpace=DeleteCases[RowReduce[Transpose[m2]],
   _?(Union[#]=={0}&)],
 nullSpace=NullSpace[m2]}
Map[Dimensions[#]&,
 {m1,m2,rowSpace,columnSpace,nullSpace}]

```

```
{nullsp=NullSpace[Transpose[m1]],
 linearDependence=
  Solve[Thread[nullsp.{v1,v2,v3}=={0}],{v3}]}
{d1=Dimensions[m1],rowDim=d1[[1]],colDim=d1[[2]],
 m=Length[m1],n=Length[First[m1]]}
```

*Special vectors:* zero, unit, constant, random vectors, transposition of vectors, Hermitian transpose vectors, sparse vectors, scalar multiple of a unit vector, etc.

*Maple:*

```
with(linalg): with(LinearAlgebra):
V1 := <a,b,c,d,e,f>;
ZeroVector(5); UnitVector(2,5); ConstantVector(C,5);
RandomVector[column](5,generator=evalf(rand(0..10)/50));
Transpose(V1); HermitianTranspose(V1);
V2:=array(sparse,1..10); V2[4]:=a4; evalm(V2);
ScalarVector(Pi,2,5);
```

*Mathematica:*

```
{v1={a,b,c,d,e,f},n=5,k=2}
{zeroVector=Table[0,{i,1,n}],
 unitVector=Table[If[i==k,1,0],{i,1,n}],
 constantVector=Table[c,{i,1,n}],
 randomVector=Table[Random[Integer,{0,10}]/50,{i,1,n}],
 ColumnForm[v1],
 v2=SparseArray[{{i_,i_},{n_,1}}->{9,17*I},{n_,1}],
 MatrixForm[v2]}
{Transpose[v2]//MatrixForm,9*unitVector}
```

*Special matrices:* zero, identity, constant, and random matrices, transposition of matrices, Hermitian transpose matrices, sparse matrices, diagonal and band matrices, scalar multiple of an identity matrix, etc.

*Maple:*

```
with(linalg): with(LinearAlgebra):
ZeroMatrix(3,3); Matrix(3,3,shape=zero);
IdentityMatrix(3,3); Matrix(3,3,shape=identity);
ConstantMatrix(C,3); Matrix(3,3,shape=constant[C]);
RandomMatrix(3,3,generator=0..10,
            outputoptions=[shape=triangular[upper]]);
f:=(i,j)->(i+j); g:=(i,j)->(x||i||j +I*y||i||j);
A:=Matrix(3,3, f); Transpose(A);
A1:=Matrix(3,3, g); HermitianTranspose(A1);
map(evalc, HermitianTranspose(A1));
A2:=array(sparse,1..10,1..10); A2[2,3]:=a23; evalm(A2);
D1:=diag(seq(2*i,i=1..5));
V:=[1,2,3,4,5]; DiagonalMatrix(V,5,5);
BandMatrix([1,9,-1],1,5,5); ScalarMatrix(C,3,3);
```

*Mathematica:*

```
<<LinearAlgebra`MatrixManipulation`;
m1=Array[Sqrt,5]; n=5;
f[i_,j_]:=i+2*j; g[i_,j_]:=i+I*j;
a1=Table[f[i,j],{i,1,n},{j,1,n}];
a2=Table[g[i,j],{i,1,n},{j,1,n}];
Map[MatrixForm,{m1,a1,a2}]
Map[MatrixForm,
 {ZeroMatrix[2,4],IdentityMatrix[10],
  constantMatrix=Table[c,{i,1,n},{j,1,n}],
  randomMatrix=Table[Random[Integer,{0,10}]/50,
    {i,1,n},{j,1,n}],
  HilbertMatrix[n], HankelMatrix[n],
  Transpose[a1],Transpose[a2],C*IdentityMatrix[3],
  m2=SparseArray[{{i_,i_},{1, n}}->{9,17},{n,n}],
  DiagonalMatrix[m1], UpperDiagonalMatrix[f, n],
  LowerDiagonalMatrix[f, n],
  BlockMatrix[{{IdentityMatrix[n],ZeroMatrix[n,n]},
   {ZeroMatrix[n,n],UpperDiagonalMatrix[f,n]}}},
  Table[If[i-j==1,2,If[i-j==-1,3,If[i==j,1,0]]],
   {i,n},{j,n}],
  TridiagonalMatrix[f,n]]]
```

*Determinants, adjoint matrices, cofactors, minors, inverse matrices.*

*Maple:*

```
with(linalg): with(LinearAlgebra):
IM:=IdentityMatrix(3,3); f:=(i,j)->a||i||j;
A:=Matrix(3,3,f); det(A); Determinant(A);
equal(map(factor,evalm(A&*Adjoint(A))),det(A)*IM);
A1:=Matrix(3,3,(i,j)->(-1)^(i+j)*det(Minor(A,i,j)));
equal(Adjoint(A),Transpose(A1));
map(simplify,evalm(inverse(A)-Adjoint(A)/det(A)));
equal(inverse(A),map(simplify,
evalm(Adjoint(A)/det(A))));
inverse(A);
```

*Mathematica:*

```
<<LinearAlgebra`MatrixManipulation`;
m1=IdentityMatrix[3]; m2=Array[f,{3,3}];
f[i_,j_]:=a[i, j];
Map[MatrixForm,
 {Inverse[m2],Minors[m2,1],Minors[m2,2]}]
{k=Length[m2],k1=Length[Minors[m2,1]],
 k2=Length[Minors[m2,2]]}
mIJ[mat_,n_]:=Table[mat[[n-i+1,n-j+1]],{i,1,n},{j,1,n}];
cofactor[mat_,n_]:=Table[(-1)^(i+j)*
 mIJ[Minors[mat,n-1],n][[i,j]],[i,1,n],[j,1,n]];
adjoint[mat_,n_]:=Transpose[cofactor[mat,n]];
Det[m2]
Map[MatrixForm,
 {mIJ[Minors[m2,1],k1],mIJ[Minors[m2,2],k2],
 adjoint[m2,k],cofactor[m2,k]}]
Inverse[m2].m2//FullSimplify)===IdentityMatrix[3]
(m2.adjoint[m2,k])//FullSimplify]===
(Det[m2]*IdentityMatrix[3])//FullSimplify)
adjoint[m2,k]===Transpose[cofactor[m2,k]]
(Inverse[m2])//FullSimplify]===
(adjoint[m2,k]/Det[m2])//FullSimplify)
```

*Linear equations, linear systems, generation of equations from the coefficient matrix, generation of the coefficient matrix from equations, augmented matrices, exact solutions, rank, corank, approximate solutions, homogeneous linear systems, kernel.*

*Maple:*

```
with(linalg): with(LinearAlgebra):
M:=<<1,0,2,3>|<2,9,6,7>|<3,6,5,7>|<2,7,5,6>>;
V:=<1,0,2,-1>; var:=[x[1],x[2],x[3],x[4]];
sys:=GenerateEquations(M,var,V);
(A,b):=GenerateMatrix(sys,var); A.Vector(var)=b;
M1:=GenerateMatrix(sys,var,augmented=true);
LinearSolve(M1); augment(A,b); linsolve(A,b);
rank(A);
corank:=Mat->coldim(Mat)-rank(Mat); corank(A);
leastsqrs(A,b); linsolve(A,ZeroVector(4));
A1:=Matrix([[1,2,3],[4,5,6],[7,8,9]]);
kernel(A1,'dimensionK'); dimensionK;
```

*Mathematica:*

```
<<LinearAlgebra`MatrixManipulation`;
{a1={{6,1,1},{3,1,-1},{3,2,-6}}, b1={3,2,-1},
n=Length[a1], x=Array[v,n]}
Map[MatrixForm,
{NullSpace[a1],
x1=LinearSolve[a1,b1],Det[a1]]]
{a1.x1==b1,
generateEqs=Thread[a1.x==b1],
generateMat=LinearEquationsToMatrices[generateEqs,x],
Thread[generateMat[[1]].x==generateMat[[2]]]]
{augmentedMat=Transpose[Append[Transpose[a1],b1]],
rowReduceMat=RowReduce[augmentedMat],
sol1=Flatten[TakeRows[Transpose[rowReduceMat],-1]]];
rank[mat_]:=Length[mat]-Length[NullSpace[mat]];
corank[mat_]:=Dimensions[a1,1]-rank[mat];
{rank[a1],rank[a1]==MatrixRank[a1],corank[a1]}
Map[MatrixForm,
{a2={{6,1,1},{3,1,-1},{3,1,-1}},b2={3,2,2}}]
{basis=NullSpace[a2],Det[a2]}
{solParticular=LinearSolve[a2,b2],
solGeneral=k*Flatten[basis]+Flatten[solParticular]}
test=a2.x2/.x2->solGeneral//Simplify
{c1={{1,2,3,2},{0,9,6,7},{2,6,5,5},{3,7,7,6}},
d1={1,0,2,-1},n=Length[c1],vars=Array[v,n]}
```

```
{RowReduce[c1]//MatrixForm,
Solve[Thread[c1.vars]==d1],
NullSpace[c1],
Transpose[c1].Inverse[c1.Transpose[c1]]
    ==PseudoInverse[c1],
solLeastSqr=PseudoInverse[c1].d1}
```

*Elimination methods:* Gaussian elimination, Gauss–Jordan elimination, fraction-free elimination, reduced row-echelon form, reduced row-echelon form modulo  $n$ , the LU decomposition, the polar decomposition.

*Maple:*

```
with(linalg): with(LinearAlgebra):
A:=<<1,0,2,3>|<2,9,6,7>|<3,6,5,7>|<2,7,5,6>>;
B:=<1,0,2,-1>; Rank(A);
AB1:=augment(A,B); AB2:=<A|B>;
equal(AB1,AB2);
G1:=gausselim(AB1,'Rank_AB1','Det_AB1');
G2:=GaussianElimination(A);
Rank_AB1; Det_AB1; X:=backsub(G1);
ffgausselim(AB2,'Rank_AB2','Det_AB2');
Rank_AB2; Det_AB2;
ReducedRowEchelonForm(AB2);
IM:=IdentityMatrix(4,4); AIM:=<A|IM>;
GJ:=gaussjord(AIM);
A_inverse:=submatrix(GJ,1..4,5..8);
equal(multiply(A,A_inverse),IM);
```

*Mathematica:*

```
<<LinearAlgebra`MatrixManipulation`;
{a1={{1,2,3,2},{0,9,6,7},{2,6,5,5},{3,7,7,6}}, 
 b1={1,0,2,-1}, n=MatrixRank[a1]}
augmentedMat=Transpose[Insert[Transpose[a1],b1,n+1]];
{rowReduceMat=RowReduce[augmentedMat],
 rowReduceMat[[All,n+1]],
 a1.rowReduceMat[[All,n+1]]==b1}
eliminationGaussian[a_,b_]:= 
Last/@RowReduce[Flatten/@Transpose[{a,b}]];
```

```

Map[MatrixForm,
{eliminationGaussian[a1,b1],
RowReduce[augmentedMat,Modulus->n+1],
a2=AppendRows[a1,IdentityMatrix[4]],
a3=RowReduce[a2],
aInverse=SubMatrix[a3,{1,n+1},{n,n}],
a1.aInverse==IdentityMatrix[4]]
{list1=LUDecomposition[a1],
LUBackSubstitution[list1,b1],
{lu,perm,cond}=LUDecomposition[a1],
LUMatrices[lu],
l=LUMatrices[lu][[1]],
u=LUMatrices[lu][[2]]}
Map[MatrixForm,{l,u,l[[perm]],u,l[[perm]].u}]
Map[MatrixForm,{a4={{1,0,0},{0,1,2},{4,0,3}},
{u,s}=PolarDecomposition[N[a4]]}]

```

*Normed vector spaces*, inner product, orthogonal vectors and matrices, angle between vectors, an orthogonal (orthonormal) set of vectors, the Gram–Schmidt orthonormalization process.

*Maple:*

```

with(linalg): with(LinearAlgebra):
M := <<1,0>|<0,-1>>;
V:=Vector(3,i->v||i); U:=Vector(3,i->u||i);
VectorNorm(V,2); MatrixNorm(M,2); Norm(V,2);
Normalize(V,2); DotProduct(V,U);
VectorAngle(V,U); angle(V, U);
IsOrthogonal(M); IsUnitary(M);
GramSchmidt([V,U]);
GramSchmidt([V,U],normalized=true);

```

*Mathematica:*

```

<<LinearAlgebra`Orthogonalization`
{m1={{1,0},{0,-1}},v1=Array[v,3],u1=Array[u,3],
 s1={v1,u1},a=t^2,b=t^2+1,w1=Cross[v1,u1]}
{Norm[v1],Norm[m1],Norm[m1,Infinity],
 v1.w1,u1.w1,v1/Norm[v1],u1/Norm[u1]}/.Simplify
v1.u1==>Inner[Times,v1,u1,Plus]

```

```

angle[x_,y_]:=ArcCos[x.y/(Norm[x]*Norm[y])];
angle[v1,u1]
{vec1={1,2,3},vec2={3,2,1}}
{Projection[vec1,vec2],Normalize[vec1],
 Projection[a,b,InnerProduct->
 (Integrate[#1*#2,{t,-1,1}]&)],
 Normalize[b,InnerProduct->
 (Integrate[#1*#2,{t,-1,1}]&)]}
{g=GramSchmidt[s1],GramSchmidt[s1,Normalized->False]}
Table[{g[[i]].g[[i]],g[[i]].g[[3-i]]},{i,1,2}]//Simplify

```

*Characteristic matrices and polynomials*, eigenvalues, eigenvectors, minimal polynomials, diagonalization, trace, the Cayley–Hamilton theorem.

*Maple*:

```

with(linalg): with(LinearAlgebra):
A:=<<1,0,0>|<2,5,0>|<0,6,9>>;
charmat(A,lambda); charpoly(A,lambda);
CharacteristicMatrix(A,lambda);
CharacteristicPolynomial(A,lambda);
E_vals:=eigenvals(A); E_vecs:=[eigenvects(A)];
Eigenvalues(A); Eigenvectors(A);
minpoly(A,lambda); MinimalPolynomial(A,lambda);
for i from 1 to 3 do
  X||i:= convert(op(E_vecs[i][3]),matrix); od;
X:=Matrix([X1,X2,X3]); X_inv:=map(expand,inverse(X));
D1:=map(simplify,evalm(X_inv &*& A &*& X));
trace(A);
evalm(subs(lambda=A,charpoly(A,lambda)));

```

*Mathematica*:

```

<<LinearAlgebra`MatrixManipulation`;
{m1={{1,2,0},{0,5,6},{0,0,9}}, n=Length[m1]}
characteristicMat=m1-lambda*IdentityMatrix[n]//MatrixForm
t=Solve[Det[m1-lambda*IdentityMatrix[n]]==0,lambda]
eigenVec1=Table[NullSpace[m1-t[[i,1,2]]*IdentityMatrix[n]],
 {i,1,3}]//FullSimplify
CharacteristicPolynomial[m1,x]//Factor

```

```

{eigenVal=Eigenvalues[m1], z=Eigensystem[m1],
 eigenVec2=Eigenvectors[m1]//FullSimplify}
Do[Print["EigVal ", i, " is ", z[[1, i]], " with EigVec",
 z[[2,i]]],{i,1,3}]
minimalPolynomial[a_,x_]:=Module[{n=Length[a],m},
 m=CharacteristicPolynomial[a,x]/Apply[PolynomialGCD[##]&,
 Flatten[Minors[a-x*IdentityMatrix[n],n-1]]];
 minimalPolynomial[m1,x]//Factor
{Tr[m1],
 p=Transpose[Eigenvectors[m1]],
 d=DiagonalMatrix[Eigenvalues[m1]]}
MatrixForm[m1]==
 MatrixForm[p].MatrixForm[d].MatrixForm[Inverse[p]]
m1==p.d.Inverse[p]
{p=CharacteristicPolynomial[m1,x],l=CoefficientList[p,x],
 k=Length[l],
 theoremCayleyHamilton=
 Sum[1[[i+1]]*MatrixPower[m1,i],{i,0,k-1}]//MatrixForm}

```

*Block diagonal matrices*, Jordan block, Jordan canonical form, Jordan transformation matrix, the Jordan decomposition.

*Maple*:

```

with(linalg): with(LinearAlgebra):
A:=Matrix(2,2,(i,j)->a||i||j);
B:=Matrix(3,3,(i,j)->b||i||j); DiagonalMatrix([A,B]);
n:=5; JordanBlock(lambda,n);
M:=matrix(3,3,(i,j)->i+j); jordan(M,'P'); evalm(P);
map(simplify,evalm(P^(-1)&*M&*P)); JordanForm(A);

```

*Mathematica*:

```

<<LinearAlgebra`MatrixManipulation`;
f[i_,j_]:=i*j;
Map[MatrixForm,
{a=LowerDiagonalMatrix[f,3],
 b=UpperDiagonalMatrix[f,3], c=ZeroMatrix[3],
 BlockMatrix[{{a,c},{c,b}}]}
m1={{1,2,3},{2,-1,1},{3,-2,1}}
{s,jor}=JordanDecomposition[m1,MatrixForm/@{s,jor}]

```

```
m1====s.jor.Inverse[s]
blockJordan[var_,n_]:= 
  Table[If[i-j== -1,1,If[i==j,var,0]],{i,n},{j,n}];
n=5; blockJordan[\[Lambda],n]//MatrixForm
```

## 4.4 Tensors

*Maple:*

The functions for working with tensors are contained in the package **tensor** that includes implementation of tensor calculus (elementary operations, differentiation, general relativity calculations). We discuss the most important functions.

*Tensor definition* and operations:

**create**, **entermetric**, **invert**, covariant and contravariant metric tensors (the index character is, respectively, negative and positive);  
**type**, analysis of tensor-type object;  
**get\_compts**, **get\_char**, **get\_rank**, extracting the tensor components, the index character, and the rank.

```
with(tensor):
t1_compts:=array(symmetric,sparse,1..3,1..3):
t1_compts[1,1]:=a: t1_compts[2,2]:=b: t1_compts[3,3]:=c:
t1:=create([-1,-1],eval(t1_compts));
t2:=invert(t1,'det_t1');
det_t1; type(t1, tensor_type); type(t2, tensor_type);
t1Components:=get_compts(t1); get_char(t1); get_rank(t1);
```

**Problem:** Convert a tensor to an operator.

```
with(tensor):
t1_compts:=array(symmetric,sparse,1..3,1..3):
t1_compts[1,1]:=a: t1_compts[2,2]:=b: t1_compts[3,3]:=c:
t1:=create([-1,-1],eval(t1_compts));
t1Components:=get_compts(t1);
A:=map(unapply,t1Components,a,b,c); convert(A,listlist);
```

*Mathematica*:

Tensors in *Mathematica* are a generalization of vectors and matrices. In *Mathematica*, a tensor is represented as a set of nested lists. The nesting level is the rank of the tensor. In this Section we discuss the built-in important functions and show how to define the new functions for working with tensors (creation of tensors, elementary operations of tensor algebra and calculus, general relativity calculations).

*Tensor definition* and operations:

`Table, Array`, creating a tensor;  
`TreeForm, MatrixForm`, visualizing tensors as a tree or a matrix;  
`Length, Dimensions, TensorRank, TensorQ`, analysis of tensors;  
`[[ ]]`, extracting tensor components, subtensors;  
`Signature`, defining antisymmetric tensors.

```
{Table[i1*i2,{i1,2},{i2,3}], Array[(#1*#2)&,{2,3}]}
t1={{{a,b,c},{d,e,f}},{{g,h,i},{j,k,l}}}
{TreeForm[t1],MatrixForm[t1],Length[t1],Dimensions[t1],
TensorRank[t1],TensorQ[t1]}
{n=3,t2=SparseArray[{{1,1}→a,{2,2}→b,{3,3}→c}],
MatrixForm[t2],Inverse[t2]//MatrixForm,Det[t2],
TensorQ[t2],MatrixForm[t2[[All,2]]]}
```

**Problem:** There is no built-in notation of covariant and contravariant tensor indices in *Mathematica*. In the usual notation, we have, for example, the tensors  $\Gamma_{ij}^k(z_1, z_2)$ ,  $R_{lki}^q$ ,  $\Gamma_{kji}$ . Construct the tensor notation in *Mathematica*.

```
Format[tensorForm[ts_][iLista___]]:=Module[{ind},
  ind={iLista}/.{sup→Superscript,sub→Subscript};
  SequenceForm[ts, Apply[Sequence,ind]]];
{tensorForm[Gamma][sup[k],sub[i],sub[j]][z1,z2],
 tensorForm[R][sup[q],sub[l],sub[k],sub[i]],
 tensorForm[Gamma][sub[k],sub[j],sub[i]]}]
```

*Tensor* algebra.

*Maple*:

prod, inner and outer tensor products;

lin\_com, linear combination of tensors.

```
with(tensor):
e1:=create([-1],array(1..3,[1,0,0]));
e2:=create([-1],array(1..3,[0,1,0]));
e3:=create([-1],array(1..3,[0,0,1]));
E1:=lin_com(1,prod(e1,e1),-1,prod(e2,e2));
T1:=create([1],array([x,y,z])):
for i from 1 to 2 do
  T2[i]:=create([1],array([i,5*i,9*i])) od:
C:=lin_com(x,T1,eval(T2[1]),eval(T2[2]));
```

*Mathematica*:

Inner, Outer, inner and outer tensor products;

Transpose, linear combination of tensors, transposition of the first two indices in a tensor.

```
<<LinearAlgebra`MatrixManipulation`;
{e1={1,0,0},e2={0,1,0},e3={0,0,1},v1={x,y,z}}
t1=Outer[Times,e1,e1]-Outer[Times,e2,e2]//MatrixForm
{t2=Table[{i,5*i,9*i},{i,1,2}],t2//MatrixForm,
 linearCombination=x*v1+t2[[1]]+t2[[2]]}
{t3=Array[f,{2,3,3}],Dimensions[t3]}
{t4=Transpose[t3],Dimensions[t4]}
```

**Problem:** Determine the Kronecker product.

```
<<LinearAlgebra`MatrixManipulation`;
productKronecker[a_?SquareMatrixQ,b_?SquareMatrixQ]:=
  BlockMatrix[Outer[Times,a,b]];
productKronecker[Array[f,{2,2}],Array[g,{2,2}]]//MatrixForm
```

**Problem:** Let  $V$  be the space of all skew-symmetric Hermitian  $2 \times 2$  matrices  $M_i$ , where  $\text{tr } M_i = 0$ . We can choose the basis in  $V$ ,  $B = \{e_1, e_2, e_3\}$ , and define the *Lie product*  $[v, u] = vu - uv$ , where  $v, u \in V$ . The right-hand side of this commutator can be expressed by the antisymmetric tensor in  $\mathbb{R}^3$ , Levi–Civita symbol  $\varepsilon_{ik}^j$ , according to the following relation:

$$[e_i, e_k] = \sum_{j=1}^3 \varepsilon_{ik}^j e_j, \quad i, k = 1, 2, 3.$$

Here  $\varepsilon_{ik}^j = 0$  if any index is equal to any index,  $\varepsilon_{ik}^j = 1$  or  $-1$  if  $\{i, j, k\}$  form, respectively, an even or odd permutation of  $\{1, 2, 3\}$ . Construct the Levi–Civita tensor and verify the commutator relation.

```
Map[MatrixForm, {e1=1/2*{{0,-I},{-I,0}},  
e2=1/2*{{0,-I},{1,0}},e3=1/2*{{-I,0},{0,I}},b={e1,e2,e3}]]  
productLie[v_List,u_List]:=v.u-u.v;  
tLeviCivita[i_,j_,k_]:=Module[{lc,ind,l1},  
ind={i,j,k}; l1=Union[ind,{1,2,3}];  
If[Length[l1]<3||Length[l1]>3,lc=0,lc=Signature[ind]];  
commutatorRelation[m_,n_]:=productLie[b[[m]],b[[n]]]===  
Sum[tLeviCivita[m,n,l]*b[[l]],{l,1,3}];  
Table[commutatorRelation[i,k],{i,1,3},{k,1,3}]//Simplify
```

*Tensor* differentiation.

*Maple*:

`d1metric, d2metric`, the first and second partial derivatives of metric tensors;

`partial_diff`, the partial derivative of tensors with respect to given coordinates;

`cov_diff`, the covariant derivative of tensors;

`directional_diff`, the directional derivative of tensors;

`Christoffel1, Christoffel2`, the Christoffel symbols of the first and the second kinds;

`Riemann`, the covariant Riemann curvature tensor.

```

with(tensor): coord:=[z1,z2];
t1_compts:=array(symmetric,sparse,1..2,1..2);
t1_compts[1,2]:=exp(rho(z1,z2));
t1:=create([-1,-1],eval(t1_compts));
t2:=invert(t1,'det_t1');
partial_diff(t1,coord); D1_t1:=d1metric(t1,coord);
D2_t1:=d2metric(D1_t1,coord);
Cf1:=Christoffel1(D1_t1); Cf2:=Christoffel2(t2,Cf1);
R1:=Riemann(t2,D2_t1,Cf1);
cov_diff(t1,coord,Cf2); cov_diff(R1,coord,Cf2);
t3:=create([1],array([t1_compts[1,2],t1_compts[1,2]]));
directional_diff(F(z1,z2),t3,coord);

```

*Mathematica:*

**Problem:** Determine the Christoffel symbols of the first and the second kinds.

```

sChristoffel1[j_,k_,a_,ts_]:=Module[{n=Length[ts]},
  (1/2)*(D[ts[[j,a]],var[[k]]]
  +D[ts[[k,a]],var[[j]]]
  -D[ts[[j,k]],var[[a]]])];
sChristoffel2[j_,k_,i_,ts_,tsInv_]:=Module[{n=Length[ts]},
  Sum[tsInv[[a,i]]*sChristoffel1[j,k,a,ts],
  {a,1,n}]];
{var={z1,z2},
 t1=SparseArray[{{i_,i_},{i_,j_}}->
  {0,Exp[rho[z1,z2]]},{2,2}],
 MatrixForm[t1],
 t2=Inverse[t1], Det[t2], TensorQ[t2]}
Map[MatrixForm,
 {Table[sChristoffel1[i,j,k,t1],{i,1,2},{j,1,2},{k,1,2}],
 Table[sChristoffel2[i,j,k,t1,t2],{i,1,2},
 {j,1,2},{k,1,2}]}}//Simplify

```

*General Relativity curvature tensors* in a coordinate basis.

*Maple:*

`tensorsGR`, General Relativity curvature tensors in a coordinate basis, e.g., the Christoffel symbols of the first and second kinds, the covariant Riemann curvature tensor, etc.

```

with(tensor): coord:=[t,r,theta,phi];
t1_compts:=array(symmetric,sparse,1..4,1..4);
t1_compts[1,1]:=f(r);
t1_compts[2,2]:=-g(r)/f(r);
t1_compts[3,3]:=-r^2;
t1_compts[4,4]:=-r^2*sin(theta)^2;
t1:=create([-1,-1],eval(t1_compts));
tensorsGR(coord,t1,t2,det_t1,C1,C2,R,Ric,Ric_sc,Ein,W);
display_allGR(coord,t1,t2,det_t1,C1,C2,R,Ric,Ric_sc,Ein,W);
Christophel1:=get_compts(C1); Christophel2:=get_compts(C2);
RicTen:=get_compts(Ric); EinsteinTen:=get_compts(Ein);

```

**Problem:** Important solutions of the Einstein field equations include the Schwarzschild exact solution, obtained by Karl Schwarzschild in 1915, describing the space-time geometry of empty space surrounding any spherically symmetric uncharged and nonrotating mass. Determine the Schwarzschild metric.

```

with(tensor): coord:=[t,r,theta,phi];
t1_compts:=array(symmetric,sparse,1..4,1..4);
t1_compts[1,1]:=f(r);
t1_compts[2,2]:=-g(r)/f(r);
t1_compts[3,3]:=-r^2;
t1_compts[4,4]:=-r^2*sin(theta)^2;
t1:=create([-1,-1],eval(t1_compts));
tensorsGR(coord,t1,t2,det_t1,C1,C2,R,Ric,Ric_sc,Ein,W);
RicTen:=get_compts(Ric);
dsolve({RicTen[1,1],RicTen[2,2]},{f(r),g(r)});
```

# Chapter 5

## Geometry

In this chapter we consider the most important geometric concepts in a Euclidean space  $\mathbb{R}^n$ , where  $\mathbb{R}^2$  represents the two dimensional plane and  $\mathbb{R}^3$ , the three dimensional space.

### 5.1 Points in the Plane and Space

*Maple:*

```
with(plots):
pointplot(points, ops);    pointplot3d(points,ops);
matrixplot(matrix,ops);
listplot(list,ops);        listplot3d(list,ops);
with(Statistics):           ScatterPlot(seqX,seqY,ops);
```

*Mathematica:*

```
<<Graphics`FilledPlot`; <<Graphics`Graphics`;
<<Graphics`Graphics3D`;
ListPlot[{{x1,y1},{x2,y2},...,{xn,yn}},
 PlotStyle->PointSize[d],PlotJoined->True]
FilledListPlot[{{x1,y2},{x2,y2},...,{xn,yn}}];
ListPlot3D[{{z11,...},{z21,...},...}];
ListPlot3D[{{z11,...},{z21,...},...},shades];
ListPlot3D[Table[z[i,j],{i,i1,i2},{j,j1,j2}]];
Show[Graphics3D[{PointSize[s],Point[{{x1,y1,z1}},...]}]];
ScatterPlot3D[{{x1,y1,z1},...},
 PlotJoined->True,PlotStyle->PointSize[s]];
```

**Problem:** Determine the points of intersection of the curves  $f(x) = x^3/9 + 3$  and  $g(x) = x^2/2 + 2$ .

*Maple:*

```
with(plots):
f:=x->x^3/9+3; g:=x->x^2/2+2;
L:=[f(x),g(x)];
plot(L,x=-4..4,-1..11,color=[green,blue],thickness=[3,5]);
S:=[fsolve(f(x)=g(x),x)];
for i from 1 to nops(S) do
    Y[i]:=eval(f(x),x=S[i]); eval(g(x),x=S[i]); od;
P:=[seq([S[i],Y[i]],i=1..nops(S))];
G1:=plot(L,x=-4..4,-1..11,
          color=[green,blue],thickness=[3,5]):
G2:=pointplot(P,color=red,symbol=circle,symbolsize=30):
display({G1,G2});
```

*Mathematica:*

```
{f[x_]:=x^3/9+3, g[x_]:=x^2/2+2}
Plot[{f[x],g[x]},{x,-4,4},PlotRange->{{-4,4},{-1,11}},
      PlotStyle->
       {{Green,Thickness[0.008]},{Blue,Thickness[0.015]}];
s=NSolve[f[x]==g[x],x]
{n=Length[s], y=f[x]/.s, z=g[x]/.s}
points=Table[{Flatten[s][[i,2]],y[[i]]},{i,1,n}]
g1=Plot[{f[x],g[x]},{x,-4,4},PlotRange->{{-4,4},{-1,11}},
         PlotStyle->
          {{Green,Thickness[0.008]},{Blue,Thickness[0.015]}},
         DisplayFunction->Identity];
g2=ListPlot[points,PlotStyle->{Red,PointSize[0.04]},
            DisplayFunction->Identity];
Show[{g1,g2},DisplayFunction->$DisplayFunction];
```

**Problem:** Graph the points  $(0, -1, 1)$ ,  $(0, -5, 0)$ ,  $(0, -1, 0)$ ,  $(0, 4, 2)$ .

*Maple:*

```
with(plots):
pointplot3d({[0,-1,1],[0,-5,0],[0,-1,0],[0,4,2]},
           axes=boxed,symbol=circle,symbolsize=20,color=blue);
```

*Mathematica:*

```
<<Graphics`Graphics`;
Show[Graphics3D[{PointSize[0.05], Hue[Random[]],
    Point[{0, -1, 1}], Point[{0, -5, 0}], Point[{0, -1, 0}],
    Point[{0, 4, 2}]}, BoxRatios -> {2, 2, 1},
    ViewPoint -> {5.759, 2.606, -1.580}]];
```

**Problem:** Graph points in the plane and the space.

*Maple:*

```
with(plots): n:=100:
listplot([seq([i,i^3],i=1..n)],color=blue);
pointplot([seq([i,i^3],i=1..n)],color=blue);
with(Statistics):
x1:=<seq(1..n)>; y1:=<seq(i^3,i=1..n)>;
ScatterPlot(x1,y1,title="Scatter Plot");
listplot3d([seq(seq([i,sin(2*i)+cos(3*j)],
    i=1..3),j=1..3)],shading=z);
points1:={seq(seq(seq([i,sin(i)+cos(j),sin(i)+cos(k)],
    i=1..3),j=1..3),k=1..3)}:
pointplot3d(points1,symbol=circle,symbolsize=15,
    shading=z,axes=boxed);
points2:={seq([\cos(Pi*t/10),\sin(Pi*t/10),Pi*t/10],
    t=0..40)}:
pointplot3d(points2,symbol=circle,symbolsize=15,
    shading=z,axes=boxed);
```

*Mathematica:*

```
<<Graphics`FilledPlot`;<<Graphics`Graphics3D`;
ListPlot[Table[{i,i^3},{i,-5,5}],
    PlotStyle->PointSize[0.05]];
FilledListPlot[Table[{i,Random[Real,{-5,5}]},
    {i,-5,5}]];
ListPlot3D[Table[Sin[2*i]+Cos[3*j],{i,1,3},{j,1,3}]];
list1=Table[{Sin[t],Cos[t],t},{t,0,2*Pi,0.1}];
ScatterPlot3D[list1,PlotStyle->PointSize[0.04]];
ScatterPlot3D[list1,PlotJoined->True];
```

## 5.2 Parametric Curves

*2D parametric curve*  $\{x = x(t), y = y(t)\}$ .

*Maple:*

```
plot([x(t), y(t), t = t1..t2], horzl, vert, ops);
```

```
plot([t^2*sin(t), t^3*cos(t), t=-10*Pi..10*Pi], axes=boxed);
plot[(-t)^3*cos(t), (-t)^2*sin(t), t=-8*Pi..8*Pi]);
```

*Mathematica:*

```
ParametricPlot[{x[t], y[t]}, {t, t1, t2}];
```

```
ParametricPlot[{t^2*Sin[t], t^3*Cos[t]}, {t, -10*Pi, 10*Pi}];
ParametricPlot[{(-t)^3*Cos[t], (-t)^2*Sin[t]},
{t, -8*Pi, 8*Pi}];
```

*3D parametric curve*  $\{x = x(t), y = y(t), z = z(t)\}, t \in [t_1, t_2]$ .

*Maple:*

```
with(plots); spacecurve([x(t), y(t), z(t)], t=t1..t2, ops);
```

```
with(plots):
x:=t->-1/2*cos(3*t); y:=t->-1/4*sin(3*t); z:=t->1/7*t;
spacecurve([x(t), y(t), z(t)], t=0..10*Pi, numpoints=400);
```

*Mathematica:*

```
ParametricPlot3D[{x[t], y[t], z[t]}, {t, t1, t2}];
```

```
x[t_]:=-1/4*Cos[3*t]; y[t_]:=-1/4*Sin[3*t]; z[t_]:=-1/7*t;
ParametricPlot3D[{x[t], y[t], z[t]}, {t, 0, 10*Pi},
PlotPoints->200, BoxRatios->{1, 1, 1}];
```

**Problem:** Let  $g(x, y) = \cos(2x - \sin(2y))$ . Graph the intersection of  $g(x, y)$  with the planes  $x = 5$ ,  $y = 0.5$ .

*Maple:*

```
with(plots): g:=(x,y)->cos(2*x-sin(2*y)):
plot3d(g(x,y),x=0..2*Pi,y=0..2*Pi,axes=boxed);
CE1 := spacecurve([5, t, 0], t=0..2*Pi):
CE2 := spacecurve([5, t, g(5, t)], t=0..2*Pi):
CE3 := spacecurve([t, 0.5, 0], t=0..2*Pi):
CE4 := spacecurve([t, 0.5, g(t, 0.5)], t=0..2*Pi):
display3d({CE1,CE2},axes=boxed);
display3d({CE3,CE4},axes=boxed);
spacecurve([[5,t,0], [5,t,g(5,t)], [t,0.5,0],
           [t,0.5,g(t,0.5)]},t=0..2*Pi,axes=boxed);
```

*Mathematica:*

```
g[x_,y_]:=Cos[2*x-Sin[2*y]];
Plot3D[g[x,y],{x,0,2*Pi},{y,0,2*Pi},Mesh->False];
g1=ParametricPlot3D[{5,t,0},{t,0,2*Pi},BoxRatios->{1,1,2},
                     DisplayFunction->Identity];
g2=ParametricPlot3D[{5,t,g[5,t]},{t,0,2*Pi},
                     BoxRatios->{1,1,2}, DisplayFunction->Identity];
g3=ParametricPlot3D[{t,0.5,0},{t,0,2*Pi},
                     BoxRatios->{1,1,2}, DisplayFunction->Identity];
g4=ParametricPlot3D[{t,0.5,g[t,0.5]},{t,0,2*Pi},
                     BoxRatios->{1,1,2}, DisplayFunction->Identity];
Show[{g1,g2,g3,g4},DisplayFunction->$DisplayFunction,
      ViewPoint->{-1.870,-6.519,-2.723}];
```

**Problem:** Graph  $f(x, y) = x^2 \sin(4y) - y^2 \cos(4x)$  for  $(x, y)$  on the circle  $x^2 + y^2 = 1$ .

*Maple:*

```
with(plots):
f:=(x,y)->x^2*sin(4*y)-y^2*cos(4*x);
spacecurve({[cos(t),sin(t),0],[cos(t),sin(t),
f(cos(t),sin(t))]},t=0..2*Pi,axes=boxed,numpoints=200);
```

*Mathematica:*

```
f[x_,y_]:=x^2*Sin[4*y]-y^2*Cos[4*x];
g1=ParametricPlot3D[{Cos[t],Sin[t],0},{t,0,2*Pi}];
g2=ParametricPlot3D[{Cos[t],Sin[t],f[Cos[t],Sin[t]]},
{t,0,2*Pi}];
Show[g1,g2,ViewPoint->{2.411,5.642,3.876}];
```

### 5.3 Implicitly Defined Curves

*Maple:*

```
implicitplot(f(x, y)=c, x=x1..x2, y=y1..y2);
```

```
with(plots): f:=(x,y)->x^2-x+y^2-y+2; R:=-10..10;
implicitplot(f(x,y)=21, x=R, y=R, grid=[50,50]);
```

*Mathematica:*

```
<<Graphics`ImplicitPlot`;
ImplicitPlot[f[x,y]==c, {x,x1,x2}];
ImplicitPlot[f[x,y]==c, {x,x1,x2}, {y,y1,y2}];
```

```
<<Graphics`ImplicitPlot`;
f1[x_,y_] := x^2-x+y^2-y+2;
ImplicitPlot[f1[x,y]==21,{x,-10,10},Frame->True];
ImplicitPlot[Sin[x+y]==x*Cos[x],{x,-10,10},{y,-10,10}];
```

### 5.4 Curves in Polar Coordinates

*Maple:*

```
plot([r(t),theta(t),t=t1..t2],coords=polar,ops);
with(plots):polarplot([r(t),p(t),p=p1..p2],ops);
```

*Mathematica:*

```
<<Graphics`Graphics`;
PolarPlot[f[t], {t, t1, t2}];
ParametricPlot[{r*Cos[t], r*Sin[t]}, {t, t1, t2}];
```

**Problem:** Lissajous curves are defined by the parametric equations  $x(t) = \sin(nt)$ ,  $y(t) = \cos(mt)$ ,  $t \in [0, 2\pi]$ , where  $(n, m)$  are different coprimes. Observe the forms of the *Lissajous curves* for various  $(n, m)$ .

*Maple:*

```
with(plots):
for i from 1 to 3 do
  n:=ithprime(i): m:=ithprime(i+1): x:=sin(n*t): y:=cos(m*t):
  plot([x,y,t=0..2*Pi],scaling=constrained,colour=red); od;
G:=[seq(plot([sin(ithprime(i)*t),cos(ithprime(i+1)*t),
  t=0..2*Pi],scaling=constrained,colour=blue),i=1..10)]:
display(G, insequence=true);
```

*Mathematica:*

```
Do[n=Prime[i];m=Prime[i+1];x=Sin[n*t];y=Cos[m*t];
  ParametricPlot[{x,y},{t,0,2*Pi},PlotStyle->Blue,
  PlotRange->All,Frame->True,FrameTicks->False,
  AspectRatio->1],[i,1,10]];
```

**Problem:** Plot the rose function,  $r = \sin(n\theta)$ ,  $n = 1, \dots, 5$ , Archimedes' spiral,  $r = \theta$ , Fermat's spiral,  $r^2 = \theta$ , the hyperbolic spiral,  $r = 1/\theta$ .

*Maple:*

```
with(plots): A:=array(1..5):
for i from 1 to 5 do
  A[i]:=polarplot([\sin(i*t), 1],t=0..2*Pi,thickness=[2,1],
    title=convert(n=i, string),color=[blue, red])
od:
display(A,axes=none,scaleing=constrained);
polarplot([t, t, t=0..17*Pi]);
plot([sqrt(t), t, t=0..17*Pi],coords=polar);
polarplot([1/t, t, t=0..17*Pi], -0.4..0.4);
```

*Mathematica:*

```
<<Graphics`Graphics`;
Do[PolarPlot[Sin[n*t],{t,0,2*Pi},
  PlotStyle->Blue,Frame->True,FrameTicks->False],{n,1,5}];
PolarPlot[t,{t,0,17*Pi}]; PolarPlot[Sqrt[t],{t,0,17*Pi}];
PolarPlot[1/t, {t, 0, 10*Pi}];
```

**Problem:** Find the area between the graphs,  $r_1 = 1$  and  $r_2 = \cos(3t)$ .

*Maple:*

```
with(plots):
r1:=t->1; r2:=t->cos(3*t); Rt:=0..2*Pi;
plot([r1(t),r2(t)],t=Rt,view=[Rt,-1..1]);
polarplot([r1(t),r2(t)],t=Rt,view=[-1..1,-1..1]);
Circ := Pi; Rose := 3/2*int(r2(t)^2,t=-Pi/6..Pi/6);
AreaG := evalf(Circ-Rose);
```

*Mathematica:*

```
<<Graphics`Graphics`;
r1[t_]:=1; r2[t_]:=Cos[3*t];
Plot[{r1[t],r2[t]},{t,0,2*Pi},PlotRange->{{0,2*Pi},{-1,1}},
  PlotStyle->Blue];
PolarPlot[{r1[t],r2[t]},{t,0,2*Pi},
  PlotRange->{{-1,1},{-1,1}},PlotStyle->Hue[0.5]];
{circ=Pi,rose=3/2*Integrate[r2[t]^2,{t,-Pi/6,Pi/6}],
 areaG=N[circ-rose]}
```

## 5.5 Secant and Tangent Lines

**Problem:** Let  $f(x) = x^3 - 4x^2 + 5x - 2$ . Plot  $f(x)$ , the secant line passing through some two points, for instance,  $(b, f(b))$ ,  $(b+t, f(b+t))$ , for various values of  $t$  and  $b = 1$ ; and the tangent line passing through some point, for instance,  $(a, f(a))$ , for various values of  $a$ .

*Maple:*

```
with(plots):
f:=x->x^3-4*x^2+5*x-2; b:=1;
SL:=(x,t)->((f(b+t)-f(b))/t)*(x-b)+f(b);
TL:=(x,a)->D(f)(a)*(x-a)+f(a);
```

```

G1:=animate(f(x),x=0..4,t=0.1..2, frames=100,
            view=[0..3,-2..2], thickness=10, color=plum):
G2:=animate(SL(x,t),x=0..4,t=0.1..2,frames=100,
            view=[0..3,-2..2], thickness=5,color=green):
G3:=animate(TL(x,a),x=0..4,a=0.1..2.9,frames=100,
            view=[0..3,-2..2], thickness=5,color=blue):
display([G1, G2, G3]):
with(Student[Calculus1]):
for a from 0.1 to 2.9 by 0.5 do
    Tangent(f(x),x=a);
od;
Tangent(f(x),x=0.1,output=plot);
Tangent(f(x),x=0.1,output=plot,showpoint=false,
         tangentoptions=[color=blue,thickness=5]);

```

*Mathematica:*

```

f[x_]:=x^3-4*x^2+5*x-2; b=1;
secL[x_,t_]:=((f[b+t]-f[b])/t)*(x-b)+f[b];
tanL[x_,t_]:=f[t]+f'[t]*(x-t);
g1=Plot[f[x],{x,0,4},
        PlotStyle->{Hue[0.9],Thickness[0.02]},
        PlotRange->{{0,3},{-2,2}},DisplayFunction->Identity];
g2=Table[Plot[secL[x,t],{x,0,4},PlotStyle->{Hue[0.7]},
            PlotRange->{{0,3},{-2,2}},DisplayFunction->Identity],
         {t,0.1,2,0.1}];
g3=Table[Plot[tanL[x,t],{x,0,4},PlotStyle->{Hue[0.5]},
            PlotRange->{{0,3},{-2,2}},DisplayFunction->Identity],
         {t,0.1,2,0.1}];
Show[{g1,g2},DisplayFunction->$DisplayFunction];
Show[{g1,g3},DisplayFunction->$DisplayFunction];

```

## 5.6 Tubes and Knots

*Maple:*

*Tubes and knots* around 3D parametric curve.

```

with(plots): tubeplot([x(t),y(t),z(t)],t=t1..t2,
                      radius=r,tubepoints=m,numpoints=n,ops);

```

```
with(plots): tubeplot([2*sin(t),cos(t)-sin(2*t),cos(2*t)],  
t=0..2*Pi,axes=boxed, radius=0.25, numpoints=100,  
scaling=constrained);  
tubeplot([-2*cos(t)+2*cos(2*t)+2*sin(2*t),  
-2*cos(2*t)+2*sin(t)-2*sin(2*t), -2*cos(2*t)], t=0..2*Pi,  
axes=boxed, radius=0.25, numpoints=100, color=gold);
```

*Mathematica:*

In *Mathematica*, there is no built-in function for plotting tubes and knots. We define the functions *tubePlot1*, *tubePlot2* (according to the formulas of differential geometry and linear algebra) and show how to plot them.

```
tubePlot1[curva_List,r_,{t_,t1_,t2_},opts___]:=Module[  
{tangente, binormal, normal, theta},  
 tangente=D[curva,t]/Sqrt[D[curva,t].D[curva,t]];  
 binormal=Cross[D[curva,t],D[curva,{t,2}]])/Sqrt[  
 Cross[D[curva,t],D[curva,{t,2}]].  
 Cross[D[curva,t],D[curva,{t,2}]]];  
 normal=Cross[binormal,tangente];  
 ParametricPlot3D[curva+r*Cos[theta]*normal  
 +r*Sin[theta]*binormal//Evaluate,  
 {t,t1, t2},{theta,0,2*Pi},opts]];  
<<LinearAlgebra`Orthogonalization`;  
tubePlot2[curva_List,r_,intervalo_List,opts___]:=Module[  
 {perps,var=First[intervalo],theta},  
 perps=GramSchmidt[NullSpace[{D[curva,var]}]];  
 ParametricPlot3D[Evaluate[  
 curva+r*Transpose[perps].{Sin[theta],Cos[theta]}],  
 Evaluate[intervalo],{theta,0,2*Pi},opts];]  
{curva1={-2*Cos[t]+2*Cos[2*t]+2*Sin[2*t],  
 -2*Cos[2*t]+2*Sin[t]-2*Sin[2*t],-2*Cos[2*t]},  
 curva2={2*Sin[t],Cos[t]-Sin[2*t],Cos[2*t]}}  
tubePlot1[curva1,0.25,{t,0,2*Pi},AspectRatio->1,  
 PlotPoints->{40,20}];  
tubePlot2[curva2,0.25,{t,0,2*Pi},PlotPoints->{50,20},  
 AspectRatio->1,LightSources->{{{0.9,0.9,0.9},  
 RGBColor[1,0,0]},{{{.9,.9,.9},RGBColor[0,1,0]},  
 {{0,0,0},RGBColor[0,0,1]}},ViewPoint->{1.5,2.5,2.5}];
```

## 5.7 Surfaces in Space

*Parametrically defined surfaces,  $x = x(u, v)$ ,  $y = y(u, v)$ ,  $z = z(u, v)$ , and implicitly defined surfaces,  $f(x, y, z) = c$ .*

*Maple:*

```
with(plots):
plot3d([x(u,v),y(u,v),z(u,v)],u=u1..u2,v=v1..v2);
implicitplot3d(f(x,y,z)=c,x=x1..x2,y=y1..y2,z=z1..z2);
```

```
with(plots): F:=(x,y,z)->-x^2-2*y^2+z^2-4*y*z;
implicitplot3d({F(x,y,z)=-100,F(x,y,z)=0,
F(x,y,z)=100},x=-6..6,y=-6..6,z=-6..6);
```

*Mathematica:*

```
ParametricPlot3D[{x[u,v], y[u,v], z[u,v]}, {u, u1, u2},
{v, v1, v2}];
<<Graphics`ContourPlot3D`;
ContourPlot3D[f[x,y,z], {x, x1, x2}, {y, y1, y2}, {z, z1, z2},
Contours -> {c1, c2, ...}];
```

```
<<Graphics`ContourPlot3D`;
F[x_,y_,z_]:=-x^2-2*y^2+z^2-4*y*z;
ContourPlot3D[F[x,y,z],{x,-6,6},{y,-6,6},{z,-6,6},
Contours->{-100,0,100}];
```

**Problem:** Plot the ellipsoid  $\frac{x^2}{16} + \frac{y^2}{4} + z^2 = 1$ , the hyperboloid of one sheet  $\frac{x^2}{16} + \frac{y^2}{4} - z^2 = 1$ , and the hyperboloid of two sheets  $\frac{x^2}{9} - \frac{y^2}{4} - \frac{z^2}{4} = 1$ .

*Maple:*

```
with(plots):
x:=(u,v)->3*cos(u)*cos(v);
y:=(u,v)->2*cos(u)*sin(v); z:=(u,v)->sin(u);
```

```

plot3d([x(u,v),y(u,v),z(u,v)],u=-Pi/2..Pi/2,v=-Pi..Pi,
       axes=boxed, scaling=constrained, orientation=[58,60]);
x:=(u,v)->4*sec(u)*cos(v); y:=(u,v)->2*sec(u)*sin(v);
z:=(u,v)->tan(u);
plot3d([x(u,v),y(u,v),z(u,v)],u=-Pi/4..Pi/4,v=-Pi..Pi,
       axes=boxed, orientation=[10, 72], color=green);
implicitplot3d(x^2/9-y^2/4-z^2/4=1,x=-10..10,y=-8..8,
                z=-5..5,axes=normal,numpoints=500,orientation=[63,43]);

```

*Mathematica:*

```

x[u_,v_]:=3*Cos[u]*Cos[v];
y[u_,v_]:=2*Cos[u]*Sin[v]; z[u_,v_]:=Sin[u];
ParametricPlot3D[{x[u,v],y[u,v],z[u,v]},
{u,-Pi/2,Pi/2}, {v,-Pi,Pi}];
x[u_,v_]:=4*Sec[u]*Cos[v]; y[u_,v_]:=2*Sec[u]*Sin[v];
z[u_,v_]:=Tan[u];
ParametricPlot3D[{x[u,v],y[u,v],z[u,v]},
{u,-Pi/4,Pi/4},{v,-Pi,Pi}];
<<Graphics`ContourPlot3D`;
ContourPlot3D[x^2/9-y^2/4-z^2/4,{x,-10,10},{y,-8,8},
{z,-5,5},Contours->{1}];

```

## 5.8 Level Curves and Surfaces

*Maple:*

```

with(plots):
contourplot(f(x,y),x=x1..x2,y=y1..y2);
contourplot3d(f(x,y,z),x=x1..x2,y=y1..y2);

```

```

with(plots): f:=(x,y)->x^2-x+y^2-y+2; R:=-10..10;
contourplot(f(x,y), x=R, y=R);
contourplot3d(f(x,y),x=R,y=R,orientation=[-91,3]);
contourplot(f(x,y),x=R,y=R,grid=[50,50],axes=boxed,
            contours=10, filled=true, coloring=[green, black]);

```

*Mathematica:*

```
<<Graphics`ContourPlot3D`;
ContourPlot[f[x,y],{x,x1,x2},{y,y1,y2}];
list1=Table[f[x,y],{x,x1,x2},{y,y1,y2}];
list2=Table[f2[x,y,z],{z,z1,z2},{y,y1,y2},{x,x1,x2}];
ListContourPlot[list1];      ListContourPlot3D[list2];
ContourPlot3D[f[x,y,z],{x,x1,x2},{y,y1,y2},{z,z1,z2}];
```

for more detail, see `Options[ContourPlot]`.

```
<<Graphics`ImplicitPlot`;
<<Graphics`ContourPlot3D`;
f1[x_,y_]:=x^2-x+y^2-y+2;
ContourPlot[f1[x,y],{x,-10,10},{y,-10,10}];
ContourPlot[f1[x,y],{x,-10,10},{y,-10,10},Contours->5];
ContourPlot[f1[x,y],{x,-10,10},{y,-10,10},
            ContourLines->False];
ContourPlot[f1[x,y],{x,-10,10},{y,-10,10},
            Contours->{10,20,30},ContourShading->False];
list1=Table[Random[Real,{1,10}],{x,1,10},{y,1,10}];
ListContourPlot[list1,MeshRange->{{{-8,8},{{-8,8}}}};
list2=Table[f1[x,y],{x,1,10},{y,1,10}];
ListContourPlot[list2,MeshRange->{{{-8,8},{{-8,8}}}};
f2[x_,y_,z_]:=-x^2-2*y^2+z^2-4*y*z;
ContourPlot3D[Evaluate[f2[x,y,z]},{x,-2,2},{y,-2,2},
               {z,-2,2},Contours->{0,8},BoxRatios->{2,1,1},
               ViewPoint->{5.741,2.536,2.197}];
list3=Table[f2[x,y,z],{z,-2,2},{y,-2,2},{x,-2,2}];
ListContourPlot3D[Evaluate[list3,Axes->True,MeshRange->
                           {{-1,1},{-1,1},{-1,1}},Contours->{0,8}]];
```

**Problem:** Plot the function  $h(x, y) = \frac{x - y}{x^2 + y^2}$  and some level curves,  $(x, y) \in [-4, 4] \times [-4, 4]$ .

*Maple:*

```
with(plots):
h:=(x,y)->(x-y)/(x^2+y^2); R:=-4..4; C:=[-2,-4,-6];
plot3d(h(x,y),x=R,y=R,grid=[20,20],orientation=[15,67]);
contourplot(h(x,y),x=R,y=R,grid=[50,50],
            axes=boxed,contours=C);
```

*Mathematica:*

```

h[x_,y_]:=(x-y)/(x^2+y^2);
Plot3D[h[x,y],{x,-4,4},{y,-4,4},PlotPoints->{20,20}];
ContourPlot[Evaluate[h[x,y]},{x,-4,4},{y,-4,4},
Contours->20,ContourShading->False,PlotPoints->{80,80}]];

```

## 5.9 Surfaces of Revolution

A *surface of revolution* is a surface obtained by rotating a curve lying on some plane,  $z = f(x)$  (or a parametric curve  $x = x(t)$ ,  $z = z(t)$ ), around a straight line (the axis of rotation) that lies on the same plane ( $z$ -axis).

In *Maple*, there is no built-in function for plotting surfaces of revolution, but we show how to plot them.

In *Mathematica*, the function for plotting surfaces of revolution is defined as

```

<<Graphics`SurfaceOfRevolution`;
SurfaceOfRevolution[f[x],{x,x1,x2},
                    RevolutionAxis->{x,y,z}];
SurfaceOfRevolution[f[x],{x,x1,x2},{t,t1,t2}];
SurfaceOfRevolution[{x[t],z[t]},{t,t1,t2}];

```

**Problem:** Plot the surface of revolution created by revolving the region bounded by the graphs of  $f(x) = x^2$ ,  $x = 0$ ,  $x = 1$ , and the  $x$ -axis around the  $y$ -axis.

*Maple:*

```

f:=x->x^2; F:=[x*cos(t),x*sin(t),f(x)];
plot3d(F,x=0..1,t=0..2*Pi);

```

*Mathematica:*

```

<<Graphics`SurfaceOfRevolution`;
SurfaceOfRevolution[x^2,{x,0,1},
                    RevolutionAxis->{0,0,1},BoxRatios->{1,1,1}];

```

## 5.10 Vector Fields

*2D and 3D vector fields* and gradient fields.

*Maple:*

```
with(plots):
fieldplot([f(x,y),g(x,y)],x=x1..x2,y=y1..y2);
gradplot(f(x,y),x=x1..x2,y=y1..y2);
fieldplot3d([f(x,y,z),g(x,y,z),h(x,y,z)],
           x=x1..x2,y=y1..y2,z=z1..z2);
gradplot3d(f(x,y,z),x=x1..x2,y=y1..y2,z=z1..z2);
```

```
with(plots): Rx:=-5..5; Ry:=-4*Pi..4*Pi;
contourplot(x*cos(y)+4*x-sin(y),x=Rx,y=Ry,grid=[50,50]);
fieldplot([x*sin(y)+cos(y),cos(y)+4],x=Rx,y=Ry,
          grid=[30,30],arrows=slim, color=x);
gradplot((1+2*x^2)*(y-1)/(1+2*y^2),x=-1..1,y=-1..1,
          grid=[20,20], arrows=thick, color=y);
fieldplot3d([x-20*y+20*z,x-4*y+20*z,x-4*y+20*z],x=-4..4,
            y=-8..8,z=-8..8,grid=[9,9,9],
            scaling=constrained,axes=boxed);
gradplot3d(10*x^2-5*y^2+2*z^2-10,x=-1..1,y=-1..1,z=-1..1,
           axes=boxed,grid=[7,7,7]);
```

*Mathematica:*

```
<<Graphics`PlotField`;
PlotVectorField[{xF,yF},{x,x1,x2},{y,y1,y2}];
<<Graphics`PlotField3D`;
PlotVectorField3D[{xF,yF,zF},
{ x,x1,x2},{y,y1,y2},{z,z1,z2}];
```

```
<<Graphics`PlotField`;
ContourPlot[x*Cos[y]+4*x-Sin[y],{x,-5,5},{y,-4*Pi,4*Pi}];
PlotVectorField[{x*Sin[y]+Cos[y],Cos[y]+4},
{x,-5,5},{y,-4*Pi,4*Pi},Axes->Automatic,
PlotStyle->Hue[0.5],HeadLength->0,AspectRatio->1];
```

```
<<Graphics`PlotField3D`;
PlotVectorField3D[{x-20*y+20*z,x-4*y+20*z,x-4*y+20*z},
{y,-8,8},{z,-8,8}];
```

## 5.11 Cylindrical Coordinates

*Maple:*

```
with(plots): Rtheta:=theta1..theta2; Rr:=r1..r2;
cylinderplot(f(r,theta), theta=Rtheta, r=Rr);
cylinderplot([r(theta),theta,r],theta=Rtheta,r=Rr);
```

```
with(plots): f:=(r,theta)->cos(r)-(2+sin(4*theta));
cylinderplot(f(r,theta),theta=0..2*Pi,r=0..4*Pi,grid=[50,50]);
```

*Mathematica:*

```
<<Graphics`ParametricPlot3D`;
CylindricalPlot3D[f[r,phi],{r,r1,r2},{phi,phi1,phi2}];
```

```
<<Graphics`ParametricPlot3D`;
f[r_,phi_]:=Cos[r]-4*Sin[2*phi];
CylindricalPlot3D[Evaluate[f[r,phi]],[phi,0,2*Pi],{r,0,2*Pi}]];
```

## 5.12 Spherical Coordinates

*Maple:*

```
with(plots): Rtheta:=theta1..theta2; Rr:=r1..r2;
sphereplot(f(z,r),z=z1..z2,r=Rr);
sphereplot([z(theta),theta,z],z=z1..z2,theta=Rtheta);
```

```
with(plots):
sphereplot(2*sin(z)*cos(r)-r,r=0..2*Pi,z=0..Pi,
axes=boxed,grid=[40,40],orientation=[0,0]);
```

*Mathematica:*

```
<<Graphics`ParametricPlot3D`;
SphericalPlot3D[f[z,r],{z,z1,z2},{r,r1,r2}];

<<Graphics`ParametricPlot3D`;
SphericalPlot3D[Evaluate[1+Sin[2*z]*Sin[2*r],
{z,0,Pi},{r,0,Pi}]];
```

## 5.13 Standard Geometric Shapes

*Standard geometric 2D and 3D shapes* and their transformations.

*Maple:*

Standard geometric shapes (e.g., circles, disks, points, lines, rectangles, polygons, cuboids, cylinders, spheres, cones) and their transformations are constructed with the packages `geometry`, `geom3d`, `plottools`.

```
with(geometry): with(plottools): with(plots):
Ops:=color=red,linestyle=3:
G1:=circle([1,1],1,color=blue):
G2:=line([-1,-1],[1,1],Ops):
G3:=line([1,1],[1,0],Ops):
G4:=line([1,0],[-1,-1],Ops):
G5:=sphere([0,0,0],1):
display({G1,G2,G3,G4},scaling=constrained);
display({cuboid([0,0,0],[1,2,3]),cylinder([0,0,-1],1,4)},
       style=patchcontour,scaling=constrained,
       orientation=[-60,60]);
display({sphere([0,0,0],1),cone([0,0,1],1,-2)},
       style=wireframe,scaling=constrained,
       orientation=[-60,60]);
display(rotate(G5,0,Pi,0),axes=boxed);
display([G5,translate(G5,1,2,3)],orientation=[-50,80]);
display([G5,translate(scale(G5,0.5,0.5,0.5),1,1,1)],
       orientation=[-50,80]);
```

*Mathematica:*

Standard geometric shapes and their transformations can be constructed with the functions `Graphics[shape]`, `Graphics3D[shape]` and the package `Graphics`Shapes``.

```
<<Graphics`Shapes`;
g1=Graphics[Circle[{1,1},1]];
g2=Graphics[Line[{{{-1,-1},{1,1},{1,0},{-1,-1}}}];
Show[{g1,g2},AspectRatio->Automatic];
Show[Graphics3D[{Cuboid[{0,0,0},{1,2,3}],
    Cylinder[1,2,30]}]];
Show[WireFrame[Graphics3D[{Sphere[1,30,30],
    Cone[1,1,30]}]];
g3=Sphere[]; g4=Graphics3D[g3];
Show[g4,RotateShape[g4,0,Pi,0]];
Show[g4,TranslateShape[g4,{1,2,3}]];
Show[g4,TranslateShape[
    AffineShape[g4,{0.5,0.5,0.5}],{1,1,1}]];
```

# Chapter 6

## Calculus

### 6.1 Differential Calculus

In Maple, most of calculus functions are contained in the packages: `student`, `Student`, `VectorCalculus`. The package `Student` contains the subpackages: `Calculus1`, `LinearAlgebra`, `Precalculus`, `VectorCalculus`, and `MultivariateCalculus`, with functions covering the basic material of the corresponding course. With the package `VectorCalculus` can be performed multivariate and vector calculus operations.

The *limit* of the function  $f(x)$  when  $x$  tends to  $x_0$ , the *derivatives* of  $f(x)$  with respect to  $x$ , the total differential of a function, the relative minimum/maximum of  $f(x)$  near  $x_0$ .

Maple:

```
limit(f(x),x=x_0);diff(f(x),x); Diff(f(x),x);D(f)(x);
diff(f(x),x$n); Diff(f(x),x$n); (D@@n)(f)(x);
PDEtools[declare](x(t),y(t),Dt=t); diff(f(x(t),y(t)),t);
minimize(f(x),x=a..b,location=true);maximize(f(x),x=a..b);
```

Mathematica:

```
Limit[f[x],x->x0] Limit[f[x],x->x0,Direction->1]
f'[x] f''[x] f'''[x] f''''[x] ... Sqrt' Sin'
D[f[x], x] D[f[x], {x,n}] Dt[f]
Derivative[n] Derivative[n][f] Derivative[n][f][x]
FindMinimum[f[x],{x,x0}] -FindMinimum[-f[x],{x,x0}]
```

**Problem:** Graph the function  $f(x) = \frac{x^3 - 3x}{x^3 - x}$ ,  $x \in [-2, 2]$ , and evaluate

the limits  $\lim_{x \rightarrow 0} f(x)$ ,  $\lim_{x \rightarrow \pm 1} f(x)$ ,  $\lim_{x \rightarrow +\infty} f(x)$ ,  $\lim_{x \rightarrow 0} \sin(1/x)$ .

*Maple:*

```
f:=x->(x^3-3*x)/(x^3-x);
plot(f(x),x=-2..2,-100..100,discont=true);
limit(f(x),x=0); limit(f(x),x=1); limit(f(x),x=-1);
limit(f(x),x=infinity); limit(sin(1/x),x=0);
```

*Mathematica:*

```
f[x_]:=(x^3-3*x)/(x^3-x); Plot[f[x],{x,-2,2},Frame->True];
{Limit[f[x],x->0], Limit[f[x],x->Infinity]}
{Limit[f[x],x->1,Direction->-1],
 Limit[f[x],x->-1,Direction->1], Limit[Sin[1/x],x->0]}
```

**Problem:** Calculate the derivatives of the functions:  $g(x) = \frac{\cos^2(x^2+1)}{(\sin x+1)^2}$ ,  $f(x) = \sqrt{x-2}(2x^2+1)^3$ ,  $f(g(x))$ , and  $g(f(x))$ .

*Maple:*

```
g:=x->cos(x^2+1)^2/((sin(x)+1)^2; f:=x->sqrt(x-2)*(2*x^2+1)^3;
D(f)(x); diff(g(x),x); diff(f(g(x)),x); diff(g(f(x)),x);
```

*Mathematica:*

```
g[x_]:=Cos[x^2+1]^2/(Sin[x]+1)^2; f[x_]:=Sqrt[x-2]*(2*x^2+1)^3;
{f'[x],g'[x],D[f[g[x]],x],D[g[f[x]],x]}/.Simplify
{Derivative[1][f][x],Derivative[1][g][x]}/.Simplify
```

**Problem:** Let  $f(x) = x^3 - 4x^2 + 8x - 2$ . Calculate  $f'(x)$ ,  $f'(\frac{7}{3})$  and find the value of  $x$  for which  $f'(x) = 10$ .

*Maple:*

```
f:=x->x^3-4*x^2+8*x-2; D(f)(x); D(f)(7/3); solve(D(f)(x)=10,x);
```

*Mathematica:*

```
f[x_]:=x^3-4*x^2+8*x-2; g:=Derivative[1][f];
{f'[x], D[f[x],x]/.x->7/3, g[x], g[7/3]}
Solve[f'[x]==10,x]
```

**Problem:** Let  $f(x) = (5x - 2)^2(2 - 5x^2)^3$ . Calculate  $f'(x)$ . Find the values of  $x$  for which the tangent line of  $f(x)$  at  $(x, f(x))$  is horizontal.

*Maple:*

```
f:=x->(5*x-2)^2*(2-5*x^2)^3;
df:=diff(f(x),x); factor(df); solve(df=0,x);
```

*Mathematica:*

```
f[x_]:=(5*x-2)^2*(2-5*x^2)^3; df=f'[x]; Factor[df];
Solve[df==0,x]
```

**Problem:** Let  $f(x) = x^2 \cos^2 x$ . Calculate  $f'(x)$ ,  $f''(x)$ ,  $f^{(3)}(x)$ , and  $f^{(4)}(x)$ . Graph  $f(x)$  and their derivatives for  $x \in [-\pi, \pi]$ .

*Maple:*

```
f:=x->x^2*cos(x)^2; df1:=diff(f(x),x); df2:=diff(f(x),x$2);
df3:=diff(f(x),x$3); df4:=diff(f(x),x$4);
plot([f(x),df1,df2,df3,df4],x=-Pi..Pi,
color=[blue,green,red,plum],
linestyle=[SOLID,DOT,DASH,DASHDOT],thickness=5);
```

*Mathematica:*

```
f[x_]:=x^2*Cos[x]^2;
df1=f'[x]; df2=f''[x]; df3=f'''[x]; df4=f''''[x];
Plot[{f[x],df1,df2,df3,df4},{x,-Pi,Pi},PlotStyle->
{{Hue[0.3],Thickness[0.02]},{Hue[0.5],Thickness[0.03]},
{Hue[0.7],Thickness[0.02],Dashing[{0.03}]},{Hue[0.8],Thickness[0.02]},{Hue[0.9],Thickness[0.01],
Dashing[{0.01,0.02}]}},AspectRatio->1];
```

**Problem:** Let  $f(x) = 2x^3 - 8x^2 + 10x$ . Find  $f'(x)$  and  $f''(x)$ . Graph  $f(x)$ ,  $f'(x)$ , and  $f''(x)$ ,  $x \in [-1, 4]$ . Determine the critical and inflection points of  $f(x)$ .

*Maple:*

```
with(plots): f:=x->2*x^3-8*x^2+10*x;
df1:=diff(f(x),x); df2:=diff(f(x),x$2);
plot([f(x),df1,df2],x=-1..5,-15..20,thickness=[5,1,1],
      color=[blue,green,magenta]); factor(df1);
p_cr:=[solve(df1=0,x)]; p_inf:=solve(df2=0,x);
eval(df2,x=p_cr[1]); eval(df2,x=p_cr[2]);
for i from 1 to nops(p_cr) do
  Y_cr[i]:=eval(f(x),x=p_cr[i]); od;
Y_inf:=eval(f(x),x=p_inf);
G1:=plot([f(x),df1,df2],x=-1..5,-15..20,
          color=[blue,green,magenta],thickness=[5,1,1]):
P:=[seq([p_cr[i],Y_cr[i]],i=1..nops(p_cr)),[p_inf,Y_inf]];
G2:=pointplot(P,color=red,symbol=circle,symbolsize=20):
display([G1, G2]);
```

*Mathematica:*

```
f[x_]:=2*x^3-8*x^2+10*x; df1=f'[x]; df2=f''[x];
Plot[{f[x],df1,df2},{x,-1,5},AspectRatio->1,
      PlotStyle->{{Hue[0.5],Thickness[0.01]},{Hue[0.7],
      Thickness[0.02]},{Hue[0.9],Thickness[0.03]}]]; Factor[df1]
pcr=Solve[df1==0,x]; pinf=Solve[df2==0,x];
xcr={pcr[[1,1,2]],pcr[[2,1,2]]}; Evaluate[df2/.pcr]
ycr=Map[f,{pcr[[1,1,2]],pcr[[2,1,2]]}]
yinf=f[pinf[[1,1,2]]]
g1=Plot[{f[x],df1,df2},{x,-1,5},AspectRatio->1,PlotStyle->
  {{Hue[0.5],Thickness[0.01]},{Hue[0.7],Thickness[0.02]},
   {Hue[0.9],Thickness[0.03]}},DisplayFunction->Identity];
ps={{xcr[[1]],ycr[[1]]},{xcr[[2]],ycr[[2]]},{pinf[[1,1,2]],yinf}}
g2=ListPlot[ps,PlotStyle->{PointSize[0.03],Hue[0.4]},
            DisplayFunction->Identity];
Show[g1,g2,DisplayFunction->$DisplayFunction];
```

**Problem:** Let  $f(x) = (x - \pi)(\pi + x) \cos x$ . Plot  $f(x)$  and the tangent line of  $f(x)$  at  $t(x_0, f(x_0))$  for  $x_0 \in [0, 3\pi]$ .

*Maple:*

```
with(plots): f:=x->(x-Pi)*(Pi+x)*cos(x); df:=diff(f(x),x);
G1:=plot(f(x),x=0..3*Pi,color=blue,thickness=5):
```

```
L_tan := proc(x0) local lin, G2;
lin:=subs(x=x0,df)*(x-x0)+f(x0);
G2:=plot(lin,x=0..3Pi,color=green,thickness=3):
display({G1, G2}); end:
for i from 0 to 10 do L_tan(i*3*Pi/10) od;
```

*Mathematica:*

```
f[x_]:=(x-Pi)*(Pi+x)*Cos[x]; df=f'[x];
tanline[x0_]:=Module[{lin,g1,g2},
lin=(df/.x->x0)*(x-x0)+f[x0];
g1=Plot[f[x],{x,0,3*Pi},PlotStyle->
{Blue,Thickness[0.02]},DisplayFunction->Identity];
g2=Plot[lin,{x,0,3*Pi},PlotStyle->
{Green,Thickness[0.01]},DisplayFunction->Identity];
Show[g1,g2,DisplayFunction->$DisplayFunction];
Do[tanline[i*3*Pi/10],{i,0,10}]
```

**Problem:** For the function  $f(x) = ax + b$  and the point  $(x_0, y_0)$  not in  $f(x)$  find the value of  $x$  such that the distance between  $(x_0, y_0)$  and  $(x, f(x))$  is minimal.

*Maple:*

```
with(student): f:=x->a*x+b;
d_min :=simplify(distance([x0,y0],[x,f(x)]));
dd_min:=simplify(diff(d_min,x));
xd_min:=solve(numer(dd_min)=0,x);
xd_min:=simplify(xd_min); yd_min:=simplify(f(xd_min));
```

*Mathematica:*

```
f[x_]:=a*x+b; mindis=ExpandAll[Sqrt[(y0-f[x])^2+(x-x0)^2]]
dmd=D[mindis,x]
{xmd=Solve[Numerator[dmd]==0,x], ymd=Simplify[f[x]/.xmd]}}
```

**Problem:** Find the dimensions of a cone of the minimal volume that is circumscribed about a sphere of radio  $R$ .

*Maple:*

```
Sol := [solve((h-R)/R=(sqrt(r^2+h^2))/r, h)];
Sol1 := remove(has, Sol, 0);
V := subs(h=op(Sol1), (Pi/3)*r^2*h);
dV := simplify(diff(V, r));
p_cr :=[solve(dV=0,r)]; p_cr1:=select(has,p_cr,R);
for i from 1 to nops(p_cr1) do
  p_cr2[i]:=p_cr1[i]/p_cr1[i]; od;
for i from 1 to nops(p_cr2) do
  if p_cr2[i]>0 then x_cr:=p_cr2[i]*p_cr1[i]; fi; od;
x_cr; ddV := simplify(diff(V,r$2));
subs(r=x_cr,ddV);  subs(r=x_cr,V);
```

*Mathematica:*

```
sol=Solve[(h-R)/R==Sqrt[r^2+h^2]/r,h]
sol1=Flatten[DeleteCases[sol,{x_>0}]]
{v=Pi/3*r^2*h/.sol1,dv=D[v,r]]//Simplify,
pCr=Solve[dv==0,r],
pCr1=Flatten[DeleteCases[pCr,{x_>0}]],
pCr2=pCr1/pCr1,n=Length[pCr2]]
Do[If[pCr2[[i]]>0,xCr=pCr2[[i]]*pCr1[[i]]],{i,1,n}]
{xCr,ddv=D[v,{r,2}]//Expand,ddv/.xCr,v/.xCr}
```

**Problem:** Calculate  $dy/dx$  if  $x^3 + y^3 = 1$ .

*Maple:*

```
ec1:=D(x^3+y^3=1); ec2:=subs(D(x)=1,ec1); isolate(ec2,D(y));
```

*Mathematica:*

```
Solve[Dt[x^3+y^3==1] /. Dt[x]->1, Dt[y]]
```

**Problem:** Determine the relative minima and maxima of the function  $f(x) = x \cos(2x)$ ,  $x \in [-\pi, \pi]$ .

*Maple:*

```
f:=x->x*cos(2*x);
plot(f(x),x=-Pi..Pi,color=blue, thickness=3);
evalf(minimize(f(x), x=-2..0, location=true));
evalf(maximize(f(x), x=-2..0, location=true));
evalf(minimize(f(x), x=0..2, location=true));
evalf(maximize(f(x), x=0..2, location=true));
```

*Mathematica:*

```
f[x_]:=x*Cos[2*x];
Plot[f[x],{x,-Pi,Pi},PlotStyle->{Blue,Thickness[0.02]}];
{FindMinimum[f[x],{x,-1}],FindMinimum[f[x],{x,1}]}
{-FindMinimum[-f[x],{x,0.1}],-FindMinimum[-f[x],{x,-1.5}]}
```

## 6.2 Integral Calculus

*Maple:*

`leftbox, rightbox, leftsum, rightsum`, graphical and numerical approximations of an integral;  
`RiemannSum`, construction of Riemann sums;  
`int, Int, value`, evaluation of indefinite and definite integrals;  
`changevar, intparts`, integration by substitution and by parts;  
`simpson, trapezoid`, approximations of definite integrals, etc.

```
with(student);with(Student);with(Student[Calculus1]);
leftbox(f(x),x=a..b,ops); rightbox(f(x),x=a..b,ops);
leftsum(f(x),x=a..b,n);   rightsum(f(x),x=a..b,n);
RiemannSum(f(x),x=a..b,ops);
int(f(x),x);  int(f(x),x=a..b,ops); I1 := Int(f(x),x);
changevar(f(x)=u,I1); intparts(I1,u); value(I1);
simpson(f(x),x=a..b,n); trapezoid(f(x),x=a..b,n);
```

*Mathematica:*

`Integrate`, evaluation of indefinite and definite integrals;  
`NIntegrate`, numerical approximations of definite integrals.

<code>Integrate[f[x],x]</code>	<code>Integrate[f[x],{x,a,b}]</code>
<code>NIntegrate[f[x],{x,a,b}]</code>	<code>Integrate[f[x],{x,a,b}]//N</code>

**Problem:** Let  $f(x) = x^3 + 5x^2 - 2x + 1$ . Approximate  $\int_0^5 f(x)dx$  by means of Riemann sums corresponding to the regular partition of  $[0, 5]$  in  $n$  subintervals.

*Maple:*

```
with(student): with(Student[Calculus1]):
f:=x->x^3+5*x^2-2*x+1;
leftbox(f(x),x=0..5,25); rightbox(f(x),x=0..5,25);
S_i:=leftsum(f(x),x=0..5,25); evala(value(S_i));
S_s:=value(rightsum(f(x),x=0..5,n));
limit(S_s, n=infinity);
SR_i:=RiemannSum(f(x),x=0..5,method=left,partition=25);
SR_s:=RiemannSum(f(x),x=0..5,method=right,partition=n);
limit(SR_s, n=infinity);
```

*Mathematica:*

```
f[x_]:=x^3+5*x^2-2*x+1; Integrate[f[x],{x,a,b}]//N
a=0; b=5; n=25; dx=(b-a)/n;
xL[k_]:=a+(i-1)*dx; Sum[f[xL[i]]*dx,{i,1,n}]//N
xR[k_]:=a+i*dx; Sum[f[xR[i]]*dx,{i,1,n}]//N
xM[k_]:=a+(i-1/2)*dx; Sum[f[xM[i]]*dx,{i,1,n}]//N
```

**Problem:** Approximate  $\int_0^{\sqrt{\pi}} e^{-x} \cos(x) dx$ . Evaluate the indefinite and definite integrals.

*Maple:*

```
f:=x->exp(-x)*cos(x); I1:=int(f(x),x=0..sqrt(Pi));
evalf(I1); F:=x->exp(2*x)*sin(2*x);factor(int(F(x),x));
simplify(int(log(x)^2/x^4,x));
G:=x->sqrt(1+x^2); int(G(x), x=0..1);
evalf(int(cos(4*x)*exp(-4*x^2), x=-2*Pi..2*Pi));
```

*Mathematica:*

```
f1[x_]:=Exp[-x]*Cos[x];
i1=Integrate[f1[x],{x,0,Sqrt[Pi]}]//N
f2[x_]:=Exp[2*x]*Sin[2*x]; Factor[Integrate[f2[x],x]]
f3[x_]:=Sqrt[1+x^2]; Integrate[f3[x],{x,0,1}]
Integrate[1/(x^2+1),{x,0,Infinity}]
Integrate[x^(2*n),{x,1,Infinity},Assumptions->{n<-1}]
Integrate[Cos[x]/(2*x),{x,-2,5},PrincipalValue->True]
NIntegrate[Cos[4*x]*Exp[-4*x^2],{x,-2*Pi,2*Pi},
WorkingPrecision->30]
g[x_]:=Integrate[f1[t],{t,1,x}];
Plot[g[x],{x,1,3}];{g'[x], g[2]//N}
```

**Problem:** Evaluate  $\int x \cos(x^2) \sin(\sin(x^2)) dx$  with  $u = \sin(x^2)$ .

*Maple:*

```
with(student):
f := x->x*cos(x^2)*sin(sin(x^2)); I1:=Int(f(x), x);
Cv:=changevar(sin(x^2)=u, I1); I2:=value(Cv);
```

*Mathematica:*

```
f[x_]:=x*Cos[x^2]*Sin[Sin[x^2]]; Integrate[f[x],x]
```

**Problem:** Evaluate  $\int_0^{\pi/8} \frac{1}{(1-\tan x)^2} dx$  by the substitution  $u=1-\tan x$ .

*Maple:*

```
with(student): f:=x->1/((1-tan(x))^2);
I1:=Int(f(x), x=0..Pi/8);
Cv:=changevar(1-tan(x)=u, I1); I2:=evalf(Cv);
```

*Mathematica:*

```
f[x_]:=1/((1-Tan[x])^2); Integrate[f[x],{x,0,Pi/8}]
```

**Problem:** Evaluate  $\int e^{-2x} \cos 2x dx$  by parts.

*Maple:*

```
with(student): f:=x->cos(2*x)*exp(-2*x); I1:=Int(f(x),x);
R1:=intparts(I1,exp(-2*x)); R2:=intparts(R1,exp(-2*x));
for i from 1 to nops(R2) do
  if type(op(i,R2),function)=true then I2:=op(i,R2); fi; od;
I2; I3:=combine(I1-I2); I4:=R2-I2;
C:=op(1,I3); CC:=remove(has,C,x); I5:=CC*I4;
```

*Mathematica:*

```
f[x_]:=Cos[2*x]*Exp[-2*x]; Integrate[f[x],x]
```

**Problem:** Let  $f(x) = \cos(\cos(\cos(x^2))) + \pi$ . Graph the function  $f(x)$  on the interval  $[\pi/2, \pi]$ , approximate  $\int_{\pi/2}^{\pi} f(x)dx$  by Simpson's and trapezoidal rules ( $n = 10$ ).

*Maple:*

```
with(student): n:=10; a:=Pi/2; b:=Pi; f:=x->(cos@@3)(x^2)+Pi;
plot(f(x),x=a..b,scaling=constrained,color=blue,thickness=5);
App_S:=simpson(f(x),x=a..b,n); V_S:=value(App_S); evalf(V_S);
App_T:=trapezoid(f(x),x=a..b,n); V_T:=value(App_T); evalf(V_T);
evalf(Int(f(x),x=a..b));
```

*Mathematica:*

```
f0[x_]:=Cos[x^2]; f1[x_]:=Nest[f0,x,3]+Pi;
Plot[f1[x],{x,Pi/2,Pi},PlotRange->All,
PlotStyle->{Hue[0.7],Thickness[0.01]}];
a=Pi/2; b=Pi; n=10; Integrate[f1[x],{x,a,b}]//N
dxTr=(b-a)/n; xTr[k_]:=a+k*dxTr;
apTr=0.5*dxTr*(f1[a]+2*Sum[f1[xTr[i]],{i,1,n-1}]+f1[b])//N
dxSim=(b-a)/(2*n); xSim[k_]:=a+k*dxSim;
apSim=dxSim/3*(f1[a]+2*Sum[f1[xSim[2*i]],{i,1,n-1}]+
4*Sum[f1[xSim[2*i-1]],{i,1,n}]+f1[b])//N
```

**Problem:** Approximate the area on the interval  $[-4, 4]$  between the curves  $p(x) = x^4 + x^3 - x^2 + x + 1$ ,  $q(x) = -0.1x^3 + 20x^2 - 10x - 20$ .

*Maple:*

```
p:=x^4+x^3-x^2+x+1; q:=-0.1*x^3+20*x^2-10*x-20;
plot([p,q],x=-4..4,color=[blue,green],thickness=3);
I_L := [fsolve(p=q, x=-4..4)];
I1 := Int(p-q, x=I_L[1]..I_L[2]);
I2 := Int(q-p, x=I_L[2]..I_L[3]); evalf(I1+I2);
```

*Mathematica:*

```
<<Graphics`FilledPlot`;
p=x^4+x^3-x^2+x+1; q=-1/10*x^3+20*x^2-10*x-20;
FilledPlot[{p,q},{x,-4,4},PlotStyle->
{{Hue[0.6],Thickness[0.01]},{Hue[0.9],Thickness[0.015]}},
{CountRoots[p-q,{x,-4,4}],intervals=RealRootIntervals[p-q]}
sols=Table[FindRoot[p==q,{x,intervals[[i,2]]}],{i,2,4}]
i1=Integrate[p-q,{x,sols[[1,1,2]],sols[[2,1,2]]}];
i2=Integrate[q-p,{x,sols[[2,1,2]],sols[[3,1,2]]}]; i1+i2//N
```

**Problem:** Let  $f(x) = \sin(\cos((x + \pi)))$ . Approximate the arc length of the graph of  $f(x)$  from  $(\pi, f(\pi))$  to  $(5\pi/3, f(5\pi/3))$ .

*Maple:*

```
f:=x->sin(cos(x+Pi));
plot(f(x),x=Pi..5*Pi/3,color=blue,thickness=3);
df:=D(f)(x); L_A:=evalf(Int(sqrt(1+df^2),x=Pi..5*Pi/3));
```

*Mathematica:*

```
f[x_]:=Sin[Cos[x+Pi]];
Plot[f[x],{x,Pi,5*Pi/3},PlotStyle->{Hue[0.6],Thickness[0.01]}];
arcL=Integrate[Sqrt[1+f'[x]^2],{x,Pi,5*Pi/3}]//N
```

**Problem:** Let  $f(x) = \cos^2 x$  and  $D$  the domain bounded by the graphs of  $y = f(x)$ ,  $x = 0$ ,  $x = \pi$ . Find the volumes of the solids obtained by revolving the bounded region  $D$  and the  $x$ -axis about the  $y$ -axis.

*Maple:*

```
f:=x->cos(x)^2; setoptions(axes=boxed);
plot(f(x),x=0..Pi,color=blue,thickness=3,scaling=constrained);
plot3d([r*cos(t),r*sin(t),f(r)],r=0..Pi,t=0..2*Pi,grid=[40,40]);
plot3d([r,f(r)*cos(t),f(r)*sin(t)],r=0..Pi,t=0..2*Pi,
grid=[40,40]); I_xy:=int(2*Pi*x*f(x), x=0..Pi);
```

*Mathematica:*

```
f[x_]:=Cos[x]^2;
Plot[f[x],{x,0,Pi},PlotStyle->{Hue[0.9],Thickness[0.01]}];
ParametricPlot3D[{r*Cos[t],r*Sin[t],f[r]},{r,0,Pi},{t,0,2*Pi}];
ParametricPlot3D[{r,f[r]*Cos[t],f[r]*Sin[t]},
{r,0,Pi},{t,0,2*Pi}]; ixy=Integrate[2*Pi*x*f[x],{x,0,Pi}]
```

### 6.3 Series

*Manipulation* of power series.

*Maple:*

- sum, computation of power series sums;
- series, Order, generalized series expansion and the environment variable;
- taylor, mtaylor, Taylor, Maclaurin, and multivariate series expansion;
- convert, polynom, the Taylor polynomials;
- coeftayl, a coefficient in the (multivariate) Taylor series;
- powerseries, formal power series package;
- inverse, multiplicative inverse of a formal power series;
- multiply, multiplication of power series, etc.

```
sum(f(i),i); sum(f(i),i=a..b); Order; series(f(x),x=a,n);
taylor(expr,x=a,n); taylor(expr,x=0,n);
mtaylor(expr,[vars],n); convert(ser,polynom);
coeftayl(expr,eqn,k); with(powseries):
inverse(expr); multiply(ser1,ser2);
```

*Mathematica:*

- Sum, computation of power series sums;
- Series, generalized (multivariate) series expansion;
- Normal, a polynomial representation of power series;

`InverseSeries`, power series inversion;  
`SeriesCoefficient`, a coefficient of power series.

<code>Sum[f[i], i]</code>	<code>Sum[f[i], {i, i1, i2}]</code>
<code>Series[f[x], {x, x0, n}]</code>	<code>Series[f[x, y], {x, x0, n}, {y, y0, m}]</code>
<code>Normal[ser]</code>	<code>SeriesCoefficient[ser, n]</code>
<code>InverseSeries[ser]</code>	<code>SeriesCoefficient[ser, {n1, n2, ...}]</code>

**Problem:** Evaluate the series sums  $\sum_{n=1}^{\infty} \frac{1}{2n^2+9n+10}$ ,  $\sum_{n=1}^{\infty} x^{kn}$ ,  $\sum_{n=1}^{10000} \frac{\cos n}{n}$ .

*Maple:*

```
sum(1/(2*n^2+9*n+10), n=1..infinity);
sum(x^(k*n), n=1..infinity); Ser:=n->cos(n)/n;
Points := [seq([i,Ser(i)],i=7000..10000)]:
plot(Points, style=POINT, color=blue, symbol=circle,
      symbolsize=10);      evalf(sum(Ser(i), i=1..10000));
```

*Mathematica:*

```
Sum[1/(2*n^2+9*n+10),{n,1,Infinity}]//Simplify
Sum[x^(k*n),{n,1,Infinity}]
ser[n_]:=Cos[n]/n;
points=Table[{i,ser[i]},{i,7000,10000}]//N;
ListPlot[points,PlotStyle->{PointSize[0.002],Blue}];
Sum[ser[i],{i,1,10000}]//N
```

**Problem:** Determine whether or not the series  $\sum_{n=1}^{\infty} \frac{n^2}{2^n}$ ,  $\sum_{n=1}^{\infty} \frac{2^n}{n!}$  converges.

*Maple:*

```
S1:=n->n^2/(2^n); S2:=n->2^n/n!;
I1:=simplify(int(S1(i), i=1..n));
limit(I1, n=infinity); int(S1(i), i=1..infinity);
evalf(sum(S1(i),i=1..1000));sum(S1(i),i=1..infinity);
Points := [seq([i, S2(i)], i=1..50)]:
plot(Points, style=POINT, color=blue, symbol=circle,
      symbolsize=20);      Fr := simplify(S2(i+1)/S2(i));
limit(Fr,i=infinity);sum(S2(i),i=1..infinity);
```

*Mathematica:*

```
ser1[n_]:=n^2/(2^n); ser2[n_]:=2^n/n!;
I1=Integrate[ser1[i],{i,1,n}]//Simplify
{Limit[I1,n->Infinity]//Simplify,
 Integrate[ser1[i],{i,1,Infinity}]}
{Sum[ser1[i],{i,1,1000}], Sum[ser1[i],{i,1,Infinity}]}//N
points=Table[{i,ser2[i]},{i,1,50}];//N;
ListPlot[points,
 PlotStyle->{PointSize[0.02],Blue},PlotRange->All];
fr=ser2[i+1]/ser2[i]//Simplify
{Limit[fr,i->Infinity], Sum[ser2[i],{i,1,Infinity}]}//N}
```

**Problem:** Find the interval of convergence of  $\sum_{n=0}^{\infty} \frac{2^{2n}}{n^2 + 2} (x^2 - 2)^n$ .

*Maple:*

```
Ser:=n->2^(2*n)/(n^2+2)*(x^2-2)^n;
C1:=simplify(Ser(n+1)/Ser(n));
C2:=limit(C1,n=infinity); C3:=[solve(abs(C2)<1,x)];
C3[1]; C3[2];
```

*Mathematica:*

```
ser[n_,x_]:=2^(2*n)/(n^2+2)*(x^2-2)^n;
c1=ser[n+1]/ser[n]//Simplify
c2=Limit[c1,n->Infinity]
c3=Reduce[Abs[c2]<1 && x\[Element]Reals, x]
```

Note that the condition  $x \in \text{Reals}$  is equivalent to  $x \in \text{Reals}$ .

**Problem:** Find the first  $n$  terms of the Taylor and Maclaurin series for  $f(x)$  about  $x=a$ . Find the Maclaurin polynomial for the function  $g(x)=\sin(x^2) \cos(x^2)/x^2$ . Graph  $g(x)$  along with its polynomial approximation on the interval  $[-\pi, \pi]$ .

*Maple:*

```
n:=10; series(f(x),x=a,n); series(f(x),x=0,n);
g:=x->sin(x^2)*cos(x^2)/x^2; Serg:=series(g(x),x=0,10);
g9 := unapply(convert(Serg,polynom),x);
plot([g(x),g9(x)], x=-Pi..Pi, -1..1,
 color=[blue,green],thickness=[5,3]);
```

*Mathematica:*

```
n=10; {Series[f[x],{x,a,n}],Series[f[x],{x,0,n}]}
g[x_]:=Sin[x^2]*Cos[x^2]/x^2; gs=Series[g[x],{x,0,7}]
g7[t_]:=Normal[gs]/.x->t;
Plot[{g[x],g7[x]},{x,-Pi,Pi},AspectRatio->1,PlotStyle->
{{Hue[0.3],Thickness[0.01]},{Hue[0.6],Thickness[0.02]}}];
```

**Problem:** Let  $f(x) = \frac{x^2}{x^2 + 2}$ . Approximate  $f(x)$  on the interval  $[0, 1]$

using the  $n$ -th degree Maclaurin polynomial and find the upper bound of the error.

*Maple:*

```
n:=6; a:=0; b:=1; f:=x->x^2/(x^2+2);
PM:=proc(k) convert(series(f(x),x=0,k),polynom); end;
df:=proc(k) simplify(eval(diff(f(x),x$(k+1)),x=y));end;
R := k -> df(k)*x^(k+1)/(k+1)!;
PM(n); df(n); R(n);
plot(df(n), y=a..b);
plot([f(x),PM(n)], x=a..b, color=[blue,green],
      thickness=[5,3]);
Sol := [evalf(solve(diff(df(n), y)=0, y))];
Cons := select(type, Sol, positive);
C_max := evalf(subs(y=op(Cons), df(n)));
Err_S := C_max*b^(n+1)/(n+1)!;
```

*Mathematica:*

```
f[x_]:=x^2/(x^2+2); {n=6,a=0,b=1}
pMaclaurin[n_]:=Module[{},Series[f[x],{x,0,n}]/.Normal];
df[n_]:=Module[{},Simplify[D[f[x],{x,n+1}]/.{x->y}]];
r[n_]:=df[n]*x^(n+1)/(n+1)!;
{pM=pMaclaurin[n],df[n],r[n]}
Plot[df[n],{y,a,b},PlotStyle->{Red,Thickness[0.01]}];
Plot[{f[x],pM},{x,a,b},PlotStyle->
     {{Blue,Thickness[0.009]},{Green,Thickness[0.01]}}];
sol=Solve[D[df[n],y]==0,y]/.N
{cons=Flatten[
  Select[sol,#[[1,2]]\[Element]Reals&&#[[1,2]]>0]};
cMax=N[df[n]/.cons], errS=cMax*b^(n+1)/(n+1)!
```

**Problem:** Let  $f(x) = \sin x$ . Construct the Taylor polynomials of degrees  $i = 1, \dots, 10$  about  $a = 1$  and compute their value at  $x = \pi/12$  and the corresponding approximation error.

*Maple:*

```
f:=x->sin(x); exval:=evalf(f(Pi/12));
apval:= n-> evalf(subs(x=Pi/12,
  convert(series(f(x),x=1,n),polynom)));
printf("      T(Pi/12)      Error");
for i from 2 to 11 do
  p1:=apval(i): p2:=evalf(abs(p1-exval));
printf(" %12.11f, %12.11f\n", p1, p2); od:
```

*Mathematica:*

```
f[x_]:=Sin[x]; exval=f[Pi/12]//N;
apval[n_]:=Normal[Series[f[x],{x,1,n}]]/.x->Pi/12//N
points=Table[{apval[i],N[Abs[apval[i]-exval]]},{i,1,10}]
TableForm[points,
  TableHeadings->{Automatic, {"T(Pi/12)", "Error"} }]
```

## 6.4 Fourier Series

In *Maple*, there is no a single function for finding an expansion of a function in terms of a set of complete functions, or one of the simplest class of the Fourier expansions, an expansion in terms of the trigonometric functions

$$1, \cos(x), \sin(x), \cos(2x), \sin(2x), \dots,$$

or their complex equivalents  $\phi_n = e^{-inx}$  ( $n = 0, \pm 1, \pm 2, \dots$ ), which are complete and orthogonal on the interval  $(-\pi, \pi)$  (or any interval of length  $2\pi$ ).

In *Mathematica*, with the package `Calculus`FourierTransform`` can be studied the Fourier exponential and trigonometric series, the corresponding Fourier coefficients (see `?Calculus`FourierTransform`*`):

```
<<Calculus`FourierTransform`
tsF[n_]:=FourierTrigSeries[Sin[t],t,n]; tsF[10]
Plot[Evaluate[tsF[10]],{t,-Pi,Pi}],PlotStyle->Blue];
```

**Problem:** For the set of variables  $x, 2x, 3x, \dots$  create the set of functions of the Fourier series.

*Maple:*

```
N:=20; var:=x; L_N:=[`i*x`$ `i'=1..N];
Fourier_sin:=convert(map(sin, L_N), set);
Fourier_cos:=convert(map(cos, L_N), set);
Fourier_F:=Fourier_sin union Fourier_cos;
```

*Mathematica:*

```
{n=20,var=x,nl=Table[i*x,{i,1,n}],sinFourier=Map[Sin,nl],
cosFourier=Map[Cos,nl],fF=Union[sinFourier, cosFourier]}
```

**Problem:** Find the Fourier coefficients of  $f = \left(a_0 + \sum_{i=1}^4 a_i \sin(ix)\right)^4$ .

*Maple:*

```
f:=(a[0]+add(a[i]*sin(i*x),i=1..4))^4; f:=combine(f,trig);
for i from 1 to 5 do
  R1:=collect(f,sin(i*x)); R2:=coeff(R1,sin(i*x)):
  print(i,"the coefficient is", R2); od:
```

*Mathematica:*

```
f=TrigReduce[(a[0]+Sum[a[i]*Sin[i*x],{i,1,4}])^4//Expand];
For[i=1,i<=5,i++, r1=Collect[f,Sin[i*x]];
r2=Coefficient[r1,Sin[i*x]];
Print[i," the coefficient is ",r2];]
```

**Problem:** Consider  $f(x) = \begin{cases} 1, & 0 \leq x \leq 1 \\ -x, & -1 \leq x < 0 \end{cases}$  with the period 2. Approximate the function  $f(x)$  by the Fourier series. Compute and graph the first 20 partial sums of the Fourier series.

*Maple:*

```
f:=proc(x) if x>=0 and x<=1 then 1 elif x<0 and x>=-1
           then -x elif x<-1 then f(x+2) fi: end;
f1:=x->-x; f2:=x->1: N:=20 ; L:=1;
a[0]:=evalf(1/(2*L)*
            (int(f1(x),x=-1..0)+int(f2(x),x=0..1)));
for i from 1 to N do
  a[i]:=evalf(1/L*(int(f1(x)*cos(i*Pi*x/L),x=-1..0)
                +int(f2(x)*cos(i*Pi*x/L),x=0..1)));
  b[i]:= evalf(1/L*(int(f1(x)*sin(i*Pi*x/L), x=-1..0)
                +int(f2(x)*sin(i*Pi*x/L), x=0..1))); od:
Term_n:=n->a[n]*cos(n*Pi*x/L)+b[n]*sin(n*Pi*x/L);
Appr_f:=n->a[0]+add(Term_n(i),i=1..n);
plot(['f(x)',Appr_f(10)],x=-10..1,color=[green,blue],
      discont=true, thickness=[2, 3], numpoints=200);
```

*Mathematica:*

```
f[x_]:=-x; -1<=x<0; f[x_]:=1; 0<=x<=1; f[x_]:=f[x+2]/;x<-1;
f1[x_]:=-x; f2[x_]:=1; n=20; L=1; a[0]=1/(2*L)*
  (Integrate[f1[x],{x,-1,0}]+Integrate[f2[x],{x,0,1}]//N;
For[i=1,i<=n,i++,
  a[i]=1/L*(Integrate[f1[x]*Cos[i*Pi*x/L],{x,-1,0}]
    +Integrate[f2[x]*Cos[i*Pi*x/L],{x,0,1}]//N;
  b[i]=1/L*(Integrate[f1[x]*Sin[i*Pi*x/L],{x,-1,0}]
    +Integrate[f2[x]*Sin[i*Pi*x/L],{x,0,1}]//N];
termN[n_]:=a[n]*Cos[n*Pi*x/L]+b[n]*Sin[n*Pi*x/L];
apprF[n_]:=a[0]+Sum[termN[i],{i,1,n}];
Plot[{f[x],apprF[10]}, {x,-10,1}, PlotStyle->\{Blue,Green\}];
```

**Problem:** Approximate the square wave waveform of period  $2\pi$ .

*Maple:*

```
a:=0; b:=2*Pi; h:=Heaviside(t-Pi);
plot(h,t=a..b,scaling=constrained);
A:=i->1/Pi*int(h*cos(i*t),t=a..b);
B:=i->1/Pi*int(h*sin(i*t),t=a..b); A0:=1/(2*Pi)*int(h,t=a..b);
F:=n->evalf(A0+sum(B(i)*sin(i*t),i=1..n));
plot(F(30),t=a..b,scaling=constrained,color=blue);
```

*Mathematica:*

```

h[t_]:=UnitStep[t-Pi];
Plot[h[t],{t,0,2*Pi},PlotStyle->Hue[0.9]];
A[i_]:=1/Pi*Integrate[h[t]*Cos[i*t],{t,0,2*Pi}];
B[i_]:=1/Pi*Integrate[h[t]*Sin[i*t],{t,0,2*Pi}];
A0=1/(2*Pi)*Integrate[h[t],{t,0,2*Pi}];
F[n_]:=A0+Sum[B[i]*Sin[i*t],{i,1,n}]/N;
Plot[Evaluate[F[30]],{t,0,2*Pi},PlotStyle->Hue[0.7]];

```

## 6.5 Multivariate and Vector Calculus

*Maple:*

*Numerous functions* for performing multivariate and vector calculus are contained in the packages `VectorCalculus`, `linalg`, `LinearAlgebra`.

`diff`, `Diff`, partial differentiation;  
`Doubleint`, `Tripleint`, `int`, `Int`, multiple and iterative integrals;  
`extrema`, relative extrema, etc.  
the package `VectorCalculus` with the vector calculus functions:  
`Gradient`, `Nabla`, `Laplacian`, `VectorField`, `Curl`, `Divergence`,  
`DotProduct`, `CrossProduct`, `Jacobian`, &x, ., etc.

```

with(student); with(linalg); with(LinearAlgebra);
f:=(x,y)->expr; extrema(expr,cond,vars);
D(f); D[i](f); D[i,j](f); D[i](D[j,i](f));
diff(f(x1,...,xn),x1,...,xn); diff(f(x1,...,xn),x1\$n);
int(...int(int(f,x1),x2)...,xn);
value(Doubleint(f(x,y), x=x1..x2, y=y1..y2));
value(Tripleint(f(x,y,z),x=x1..x2,y=y1..y2,z=z1..z2));
with(VectorCalculus); Gradient(expr,[x,y]);
Nabla(expr,[x,y]); Laplacian(expr,[x,y]);
VF:=VectorField(<x,y>,coordsys); Curl(VF);
Divergence(VF); DotProduct(<x,y>,<x,y>); <x,y>.<x,y>;
CrossProduct(v1,v2); v1&x v2; Jacobian([expr],[vars]);

```

*Mathematica:*

D, Derivative, Dt, partial derivatives and the total differential and derivative;

Integrate, NIntegrate, analytical integration (indefinite, definite, multiple) and numerical approximations of integrals;

VectorAnalysis, the package with the vector calculus functions;

VectorField, construction of vector fields (see Sect. 5.10).

```
f[x_,...,xn_]:=expr; D[f,x] D[f,{x,n}] D[f,x1,...,xn]
D[f,{x1,k1},...{xn, kn}] Derivative[k1,...,kn][f]
Dt[f[x,y]] Dt[f[x,y],x]
Integrate[f[x,y],{x,x1,x2},{y,y1,y2}]
Integrate[f[x,y,z],{x,x1,x2},{y,y1,y2},{z,z1,z2}]
NIntegrate[f[x,y,z],{x,x1,x2},{y,y1,y2},{z,z1,z2}]
<<Calculus`VectorAnalysis`; SetCoordinates[Coord];
gradf={D[f[x,y,z],x],D[f[x,y,z],y],D[f[x,y,z],z]}
Grad[f[x,y,z]] Laplacian[f[x,y,z]] Curl[f[x,y,z]]
Div[f[x,y,z]] DotProduct[vec1, vec2] vec1.vec2
```

**Problem:** Let  $f(x, y) = x^2 / \sqrt{x^2 + y^2}$ . Graph  $f(x, y)$  and find the second order partial derivatives.

*Maple:*

```
f:=(x,y)->x^2/sqrt(x^2+y^2);
plot3d(f(x,y),x=-1..1,y=-1..1,grid=[40,40],axes=boxed,
       orientation=[-40, 55], shading=Z);
simplify([diff(f(x,y),y,x),D[2,1](f)(x,y), diff(f(x,y),x,y),
          D[1,2](f)(x,y), diff(f(x,y),x$2),D[1,1](f)(x,y),
          diff(f(x,y),y$2),D[2,2](f)(x,y)]);
```

*Mathematica:*

```
f[x_,y_]:=x^2/Sqrt[x^2+y^2];
Plot3D[f[x,y],{x,-1,1},{y,-1,1},PlotPoints->{40,40}];
{D[f[x,y],y,x],D[f[x,y],x,y],D[f[x,y],{x,2}],
 D[f[x,y],{y,2}]}/.Simplify
```

---

```
f1=Derivative[1,1][f]; f2=Derivative[2,0][f];
f3=Derivative[0,2][f];
Map[Simplify, {f1[x,y],f2[x,y],f3[x,y]}]
```

**Problem:** Let  $f(x, y) = x^2 \cos(y^2)$ . Evaluate  $f_{xy}$  at the point  $(-\pi, \pi)$ , find the total differential, and total derivatives with respect to  $x$  and  $y$ .

*Maple:*

```
f:=(x,y)->x^2*cos(y^2); subs({x=-Pi,y=Pi},diff(f(x,y),x,y));
convert(D[1,2](f)(-Pi,Pi),diff);
PDEtools[declare](x(t),y(t),Dt=t);
tDif:=simplify(diff(f(x(t),y(t)),t));
TotDerX:=factor(Dt(y,x)*diff(f(x,y),y)+diff(f(x,y),x));
TotDerY:=factor(Dt(x,y)*diff(f(x,y),x)+diff(f(x,y),y));
```

*Mathematica:*

```
f[x_,y_]:=x^2*Cos[y^2];
{D[f[x,y],x,y]/.{x->-Pi,y->Pi},Derivative[1,1][f][-Pi,Pi],
 Dt[f[x,y]], Dt[f[x,y],x], Dt[f[x,y],y]}/. Simplify
```

**Problem:** For  $z = x^2y^2$ , where  $y = x^2$ , evaluate  $\frac{dz}{dx}$ , for  $x^{2k}$  evaluate the total differential.

*Maple:*

```
with(diffforms): z:=x^2*y^2;
z1:= subs(y=x^2,z); eval(subs(d(x)=1,d(z1)));
defform(k=scalar); z2:=(x)^(2*k);
z3:=simplify(subs(d(k)=0,d(z2)));
```

*Mathematica:*

```
z=x^2*y^2; {Dt[z,x]/.{y->x^2}, Dt[x^{2*k},Constants->{k}]} 
```

**Problem:** Find and classify the critical points of the function  $f(x, y) = x^4 + 10x^3 + 2x^2y^2 + xy^2$ .

*Maple:*

```
f      :=(x,y)->x^4+10*x^3+2*x^2*y^2+x*y^2;
df_x  :=diff(f(x,y), x);      df_y := diff(f(x,y),y);
sols  :=allvalues({solve({df_x=0,df_y=0},{x,y})});
P_cr  :=evalf(convert(map('union',
                           {op(sols[1]),op(sols[2])}),list));
N    :=nops(P_cr);           df_xx:=diff(f(x,y),x$2);
df_yy :=diff(f(x,y),y$2); df_xy:=diff(f(x,y),x,y);
d:=simplify(df_xx*df_yy-(df_xy)^2); L:=[x,y,df_xx,d];
Class :=array([seq(subs(P_cr[i], L), i=1..N)]);
```

*Mathematica:*

```
f[x_,y_]:=x^4+10*x^3+2*x^2*y^2+x*y^2;
dfx=D[f[x, y], x]; dfy = D[f[x, y], y];
sols=Solve[{dfx==0, dfy==0}, {x,y}]/.N
pcr=Union[sols, {}]; n = Length[pcr];
d=D[f[x,y],{x,2}]*D[f[x,y],{y,2}]-(D[f[x,y],x,y])^2//Simplify
crlist={x, y, D[f[x,y],{x,2}], d}
class=Table[crlist/.pcr[[i]],{i,1,n}]/.TableForm
PaddedForm[class,{12,5}]
```

**Problem:** Let  $f(x, y) = \exp(-2(x^2 + y^2))$ . Find an equation of the tangent plane to the graph of  $f(x, y)$  at  $(-1, 1)$ .

*Maple:*

```
with(plots):          f:=(x,y)->exp(-2*(x^2+y^2));
d_x:=D[1](f)(-1,1); d_y:=D[2](f)(-1,1);
Eq_tp:=evalf(d_x*(x+1)+d_y*(y-1)+f(-1,1));
setoptions3d(grid=[40,40],axes=boxed,orientation=[68,72]);
G1:=plot3d(f(x,y),x=-5..5,y=-5..5,shading=Z):
G2:=plot3d(Eq_tp, x=-5..5,y=-5..5,shading=zhue):
display({G1, G2});
```

*Mathematica:*

```
f[x_,y_]:=Exp[-2*(x^2+y^2)];
dx=Derivative[1,0][f][-1,1]; dy=Derivative[0,1][f][-1,1];
Eqtp=dx*(x+1)+dy*(y-1)+f[-1,1]/.N
g1=Plot3D[f[x,y],{x,-5,5},{y,-5,5},DisplayFunction->Identity];
g2=Plot3D[Eqtp,{x,-5,5},{y,-5,5},DisplayFunction->Identity];
Show[{g1,g2},DisplayFunction->$DisplayFunction,
      AspectRatio->1,ViewPoint->{4.790,-2.905,2.190}];
```

**Problem:** Use the method of Lagrange multipliers to find the maximum and minimum values of  $f(x, y) = 2x^2 - 3y^3 + 4(xy)^2$  subject to the constraint  $x^2 + 2y^2 = 1$ .

*Maple:*

```
with(student): with(plots):
f:=(x,y)->2*x^2-3*y^3+4*x^2*y^2; Cond:=(x,y)->x^2+y^2-1;
Ex:=evalf(extrema(f(x,y),Cond(x,y),{x,y},'Sol'));
sols := allvalues(Sol);
LSol:=evalf(convert(map('union',
{'op(sols[i])' $ 'i'=1..4}),list));
N := nops(LSol); L := [x, y, f(x, y)];
array([seq(subs(evalf(LSol[i]), L), i=1..N)]);
C1:=spacecurve([\cos(t),\sin(t),0],t=0..2*Pi,
    color=blue,thickness=3):
C2:=spacecurve([\cos(t),\sin(t),f(\cos(t),\sin(t))],t=0..2*Pi,
    color=magenta,thickness=5,orientation=[15,66]):
display3d([C1, C2], axes=boxed);
```

*Mathematica:*

```
f[x_,y_]:=2*x^2-3*y^3+4*x^2*y^2; g[x_,y_]:=x^2+y^2-1;
conds=Eliminate[{D[f[x,y],x]==lambda*D[g[x,y],x],
D[f[x,y],y]==lambda*D[g[x,y],y],g[x,y]==0},lambda]
points=Solve[conds]//Simplify; fVal=f[x,y]/.points
{Max[fVal], Min[fVal]}/.N
n = Length[points]; l1={x,y,f[x, y]};
l2=Table[l1/.points[[i]],{i,1,n}]//N//TableForm;
PaddedForm[l2,{12,5}]
```

**Problem:** Evaluate the integrals:

$$\int_0^1 \int_0^1 \sin(\cos xy) dx dy, \quad \int_0^\pi \int_0^\pi \int_0^1 e^{2xz} \cos(x^2 - y^2) dz dx dy.$$

*Maple:*

```
Ri1:=0..1; Ri2:=0..Pi;
i1:=int(int(sin(cos(x*y)),x=Ri1),y=Ri1); i1:=evalf(i1);
i2:=Int(Int(Int(exp(2*x*z)*cos(x^2-y^2),z=Ri1),
x=Ri2),y=Ri2); i2:=evalf(i2);
```

*Mathematica:*

```
Integrate[Sin[Cos[x*y]],{y,0,1},{x,0,1}]//N
Integrate[Exp[2*x*z]*Cos[x^2-y^2],{y,0,Pi},{x,0,Pi},{z,0,1}]//N
```

**Problem:** Find the volume of the region bounded by the graphs of the surfaces  $f(x, y) = x + y$  and  $g(x, y) = 10 - 2x^2 - 2y^2$  on the rectangle  $[-4, 4] \times [-4, 4]$ .

*Maple:*

```
with(plots): f:=(x,y)->x+y; g:=(x,y)->10-2*x^2-2*y^2;
setoptions3d(axes=boxed,grid=[40,40],orientation=[-21,53]);
Gf := plot3d(f(x,y),x=-4..4,y=-4..4,color=blue):
Gg := plot3d(g(x,y),x=-4..4,y=-4..4,color=green):
display3d({Gf,Gg});
y0:=[solve(f(x,y)=g(x,y), y)]; x0:=[solve(op(2,y0[1])=0,x)];
Vol:=Int(Int(g(x,y)-f(x,y),y=y0[2]..y0[1]),x=x0[1]..x0[2]);
evalf(Vol);
```

*Mathematica:*

```
f[x_,y_]:=x+y; g[x_,y_]:=10-2*x^2-2*y^2;
gf=Plot3D[f[x,y],{x,-4, 4},{y, -4,4},DisplayFunction->Identity];
gg=Plot3D[g[x,y],{x,-4, 4},{y, -4,4},DisplayFunction->Identity];
Show[gf,gg,DisplayFunction->$DisplayFunction];
y0=Solve[f[x,y]==g[x,y],y]; t=y0[[1,1,2,2,2]]
x0=Solve[t == 0,x];
Vol=Integrate[g[x,y]-f[x,y],{x,x0[[1,1,2]],x0[[2,1,2]]},
{y,y0[[1,1,2]],y0[[2,1,2]]}]/N
```

**Problem:** Let  $f(x, y, z) = x \sin^2(xy z)$ . Find  $\nabla f$ ,  $\nabla^2 f$ ,  $\operatorname{div}(\nabla f)$ .

*Maple:*

```
with(VectorCalculus): with(linalg):
f:=(x,y,z)->x*sin(x*y*z)^2; var:=[x,y,z];
grad1_f:=grad(f(x,y,z),var); grad2_f:=Gradient(f(x,y,z),var);
Lap_f:=Laplacian(f(x,y,z),var); Div_f:=Divergence(grad2_f,var);
```

*Mathematica:*

```
<<Calculus`VectorAnalysis`; f[x_,y_,z_]:=x*Sin[x*y*z]^2;
gradf={D[f[x,y,z],x],D[f[x,y,z],y],D[f[x,y,z],z]}
SetCoordinates[Cartesian[x,y,z]]
{Grad[f[x,y,z]],Laplacian[f[x,y,z]],Div[Grad[f[x,y,z]]]}
```

**Problem:** Let  $f(x, y, z) = -(xy)^2 \mathbf{i} + \cos^2(xyz) \mathbf{j} + \sin^2 z \mathbf{k}$ . Find  $\operatorname{curl} f$ ,  $\operatorname{div} f$ ,  $\Delta(\operatorname{div} f)$ ,  $\operatorname{grad}(\Delta(\operatorname{div} f))$ .

*Maple:*

```
with(VectorCalculus): SetCoordinates('cartesian'[x,y,z]);
f:=VectorField(<-(x*y)^2, cos(x*y*z)^2, sin(z)^2);
curl_f:=map(factor,Curl(f));div_f:=combine(Divergence(f));
lap_div_f:=combine(Laplacian(div_f));
grad_lap_div_f:=map(simplify, Gradient(lap_div_f));
```

*Mathematica:*

```
<<Calculus`VectorAnalysis`;SetCoordinates[Cartesian[x,y,z]];
f={-(x*y)^2,Cos[x*y*z]^2,Sin[z]^2}; xcurlf=Curl[f]//Factor
{Div[f], Laplacian[Div[f]]}// Simplify
Grad[Laplacian[Div[f]]]//Simplify
```

**Problem:** Let  $f(x, y) = \sin^2(xy) + 2\cos(xy)^3$ . Find the formula of a unit normal vector to the graph of  $f(x, y)$  at  $(x, y, f(x, y))$ .

*Maple:*

```
with(linalg): f:=(x,y)->sin(x*y)^2+2*cos(x*y)^3; var:=[x,y,z];
fz:=(x,y,z)->z-f(x,y); grad_f:=simplify(grad(fz(x,y,z),var));
norm_grad_f:=simplify(sqrt(dotprod(grad_f, grad_f)));
norm_f:=map(simplify,scalarmul(grad_f,1/norm_grad_f));
plot3d(f(x,y),x=-5..5,y=-5..5,axes=boxed,grid=[40,40],
shading=zhue,scaling=constrained,orientation=[-13,36]);
```

*Mathematica:*

```
<<Calculus`VectorAnalysis`
f[x_,y_]:=Sin[x*y]^2+2*Cos[x*y]^3; fz[x_,y_,z_]:=z-f[x,y];
Map[Simplify,{gradf=Grad[fz[x,y,z],Cartesian[x,y,z]],
normgradf=Sqrt [gradf.gradf],normf=gradf/normgradf}]
Plot3D[f[x,y],{x,-5,5},{y,-5,5}];
```

**Problem:** Evaluate the integral  $\oint_C (3y - e^{\sin x})dx + (7x + \sqrt{y^4 + 1})dy$ , where  $C$  is  $x^2 + y^2 = 9$ .

*Maple:*

```
with(plots): setoptions(axes=boxed, thickness=5, numpoints=400);
P:=(x,y)->3*y-exp(sin(x)); Q:=(x,y)->7*x+sqrt(y^4+1);
P_y:=diff(P(x,y),y); Q_x:=diff(Q(x,y), x); Rt:=0..2*Pi;
Fp:=[3*cos(t)]; I1:=int(int((Q_x-P_y)*r,r=0..3),theta=0..2*Pi);
G1:=polarplot(Fp,t=Rt,coords=polar,color=blue):
G2:=polarplot(Fp,t=Rt,coords=polar,color=grey,filled=true):
display([G1,G2]);
```

*Mathematica:*

```
<<Graphics`Graphics`;
p[x_,y_]:=3*y-Exp[Sin[x]];
q[x_,y_]:=7*x+Sqrt[y^4+1];
py=D[p[x,y],y];
qx=D[q[x,y],x];
i1=Integrate[(qx-py)*r,{r,0,3},{theta,0,2*Pi}]
g1=PolarPlot[{3*Cos[t]},{t,0,2*Pi},PlotStyle->
{Thickness[0.04]},DisplayFunction->Identity];
g2=Graphics[{Hue[0.8],Disk[{1.5,0},1.5]},
DisplayFunction->Identity];
Show[g1,g2,Frame->True,DisplayFunction->$DisplayFunction];
```

**Problem:** Calculate the external flow of the vector field  $F(x, y, z) = (xy + x^2yz)\mathbf{i} + (yz + xy^2z)\mathbf{j} + (xz + xyz^2)\mathbf{k}$  through the surface of the first octant bounded by the planes  $x = 2$ ,  $y = 2$ ,  $z = 2$ .

*Maple:*

```
with(linalg): F:=(x,y,z)->[x*y+x^2*y*z,y*z+x*y^2*z,x*z+x*y*z^2];
div_F:=diverge(F(x,y,z),[x,y,z]);
int(int(int(div_F,z=0..2),y=0..2),x=0..2);
```

*Mathematica:*

```
<<Calculus`VectorAnalysis`;
SetCoordinates[Cartesian[x,y,z]];
F[x_,y_,z_]:={x*y+x^2*y*z, y*z+x*y^2*z, x*z+x*y*z^2};
{divF=Div[F[x,y,z]], Integrate[divF,{z,0,2},{y,0,2},{x,0,2}]}
```

# Chapter 7

## Complex Functions

### 7.1 Complex Algebra

*Maple* performs complex arithmetic automatically. The imaginary unit  $i$  of the complex number  $x+y*I$  is denoted by  $I$ .

*Functions over complex numbers* and variables:

- `abs`, the absolute value;
- `Re`, `Im`, the real and imaginary parts;
- `conjugate`, the complex conjugate;
- `argument`, the complex argument;
- `signum`, the sign of a real or complex number;
- `csgn`, the complex “half-plane” signum function;
- `polar`, polar representation of complex numbers;
- `evalc`, complex evaluation function.

```
abs(z); Re(z); Im(z); conjugate(z); argument(z);
signum(z); csgn(z); polar(r,theta); evalc(z);
```

```
z1 := 5+I*7; abs(z1); Re(z1); Im(z1); conjugate(z1);
argument(z1); signum(z1); csgn(z1); polar(z1);
z2 := x+I*y; evalc(abs(z2)); evalc(Re(z2));
assume(x,real); assume(y,real); about(x,y); Im(z2);
unassign('x','y'); about(x,y); evalc(signum(z2));
evalc(polar(r,theta)); map(evalc,convert(z2, polar));
expand((1-I)^4); evalc(sqrt(-9)); expand((3+I)/(4-I));
```

*Mathematica* performs complex arithmetic automatically. The imaginary unit  $i$  of the complex number  $x+y*I$  is denoted by  $I$ .

*Functions over complex numbers* and variables:

**Abs**, the absolute value;

**Re**, **Im**, the real and imaginary parts;

**Conjugate**, the complex conjugate;

**Arg**, the complex argument;

**Sign**, the sign of a real or complex number;

**ComplexExpand**, complex expansion function, the polar form.

<b>Abs</b> [z]	<b>Re</b> [z]	<b>Im</b> [z]	<b>Conjugate</b> [z]	<b>Arg</b> [z]	<b>Sign</b> [z]
<b>ComplexExpand</b> [z]				<b>ComplexExpand</b> [z,{x <sub>1</sub> ,...,x <sub>n</sub> }]	

```

z1=5+I*7; z2=x+I*y;
{Abs[z1],Re[z1],Im[z1],Conjugate[z1],Arg[z1],Sign[z1]}
polar=ComplexExpand[z,{z},TargetFunctions->{Arg,Abs}]
{Abs[z2],Re[z2],Im[z2]}//ComplexExpand
{(1-I)^4,Sqrt[-9],(3+I)/(4-I)}//ComplexExpand
{ComplexExpand[Exp[-k*(a+I*c*t)],k],
 ComplexExpand[Exp[-k*(a+I*c*t)],k,
 TargetFunctions->{Abs,Arg}],
 ComplexExpand[Exp[-k*(a+I*c*t)],k,TargetFunctions->
 {Abs,Arg}]/.{Abs[k]->\[Alpha],Arg[k]->\[Phi]}}
f[theta_]:=Arg[polar/.{Abs[z]->5,Arg[z]->theta}]
Plot[f[theta],{theta,-Pi,Pi}];

```

**Problem:** Prove that  $\cos(3\theta)=\cos(\theta)^3 - 3 \cos(\theta) \sin(\theta)^2$  (use De Moivre's theorem).

*Maple:*

```

assume(theta, real); z1:=cos(3*theta)+I*sin(3*theta);
z2:=evalc((cos(theta)+I*sin(theta))^3);
R1:=Re(z1); R2:=Re(z2); R1=R2;

```

*Mathematica:*

```

z1=Cos[3*theta]+I*Sin[3*theta]
z2=(Cos[theta]+I*Sin[theta])^3//ComplexExpand
{r1=ComplexExpand[Re[z1]],r2=ComplexExpand[Re[z2]]}
r1==r2

```

**Problem:** Find and graph all the solutions of the equation  $z^9 = 1$ .

*Maple:*

```

Ops:=style=point,symbol=circle,color=blue,symbolsize=30;
Sols := map(allvalues, {solve(z^9 = 1, z)});
Points:=map(u->[Re(u),Im(u)], Sols); plot(Points,Ops);

```

*Mathematica:*

```

{sol=Solve[z^9==1,z],n=Length[sol]}
{points=sol[[Table[i,{i,1,n}],1,2]]//N,
 points=points/.{x_Real->{x,0},Complex[x_,y_]->{x,y}}}
ListPlot[points,PlotStyle->{PointSize[0.03],Hue[0.7]},
 AspectRatio->1];

```

## 7.2 Functions, Limits, Continuity

*Multivalued functions,* the principal value.

**Problem:** Find the values:  $\text{Log}(z_1)$ ,  $\log(z_2)$ ,  $z_2^c$ , where  $z_1 = 1 + 2i$ ,  $z_2 = 2$ ,  $c = (1 - i)/30$ .

*Maple:*

```

z1:=1+2*I; w1:=evalc(log(z1));
w2:=evalc(log(z1))+I*2*Pi*n;
z2:=2; c:=(1-I)/30; w3:=evalc(exp(c*log(z2)));
z3:=n->evalf(exp(c*(log(z2)+2*Pi*I*n))); k:=9;
for n from -k to k do print(z3(n)) od:
Points:=[[Re(z3(i)),Im(z3(i))] $ i=-k..k];
plot(Points,scaling=constrained,style=point,
      axes=boxed,symbol=circle,color=blue);

```

*Mathematica:*

```
{z1=1+2*I,w1=ComplexExpand[Log[z1]],  
 w2=ComplexExpand[Log[z1]]+I*2*Pi*n}  
{z2=2,c=(1-I)/30,w3=ComplexExpand[Exp[c*Log[z2]]]}  
z3[n_]:=Exp[c*(Log[z2]+2*Pi*I*n)]//N; k=9  
Do[Print[z3[n]],{n,-k,k}];  
points=Table[{Re[z3[i]],Im[z3[i]]},{i,-k,k}]  
ListPlot[points,PlotStyle->{PointSize[0.03],Hue[0.7]},  
 AspectRatio->1];
```

*Analytic and harmonic functions.*

**Problem:** Show that  $u(x, y) = x^2 - y^2$ ,  $v(x, y) = 2xy$  are harmonic functions.

*Maple:*

```
f:=z->z^2: F:= evalc(f(x+I*y));  
u:=(x,y)->evalc(Re(F)); v:=(x,y)->evalc(Im(F));  
evalb(0=diff(u(x,y),x$2)+diff(u(x,y),y$2));  
evalb(0=diff(v(x,y),x$2)+diff(v(x,y),y$2));
```

*Mathematica:*

```
f[z_]:=z^2; f1=ComplexExpand[f[x+I*y]];  
u[x_,y_]:=ComplexExpand[Re[f1]];  
v[x_,y_]:=ComplexExpand[Im[f1]];  
D[u[x,y],{x,2}]+D[u[x,y],{y,2}]==0  
D[v[x,y],{x,2}]+D[v[x,y],{y,2}]==0
```

**Problem:** Determine the harmonic conjugate  $v(x, y)$  for harmonic function  $u(x, y) = xy^3 - x^3y$ .

*Maple:*

```
VConj := proc(u)  
 local v1,v2,v3,v4;  
 v1 := int(diff(u,x),y);  
 v2 :=-diff(u,y)-diff(v1,x);v3:=int(v2,x);  
 v4 := v1+v3; RETURN(v4); end;  
 u:=(x,y)->x*y^3-x^3*y;  
 evalb(0=diff(u(x,y),x$2)+diff(u(x,y),y$2));  
 v:=(x,y)->VConj(u(x,y)); v(x,y); v(2*x,2*y);
```

*Mathematica:*

```
vConj[u_]:=Module[{v1,v2,v3,v4},
  v1=Integrate[D[u,x],y]; v2=-D[u,y]-D[v1,x];
  v3=Integrate[v2,x]; v4=v1+v3];
u[x_,y_]:=x*y^3-x^3*y;
D[u[x,y],{x,2}]+D[u[x,y],{y,2}]==0
v[x_,y_]:=vConj[u[x,y]];
{v[x,y],v[2*x,2*y]}
```

**Problem:** Find  $\lim_{z \rightarrow 1+i} \frac{z^2 - 2i}{z^2 - 2z + 2}$ .

*Maple:*

```
f:=z->(z^2-2*I)/(z^2-2*z+2); F:=factor(f(z));
L1:=subs(z=1+I, F); L2:=limit(f(z), z=1+I);
```

*Mathematica:*

```
f[z_] := (z^2-2*I)/(z^2-2*z+2);
f1=Factor[f[z1]]
{l1=f1/.{z1->1+I}, l2=Limit[f[z],z->1+I]}
```

**Problem:** Show that the function  $f(z) = z^4 - z^3 - 9z + 9$  has a removable discontinuity at  $z = 2i$ .

*Maple:*

```
with(plots): k:=2*I; f:=z->z^4-z^3-9*z+9;
F:=factor(f(z)); L:=limit(f(z),z=k);
h:=z->proc(z) if z=k then L else f(z) fi: end;
conformal('h'(z),-4*I..4*I,axes=boxed,color=blue);
```

*Mathematica:*

```
<<Graphics`ComplexMap`;
f[z_]:=z^4-z^3-9*z+9;
{k=2*I,f1=Factor[f[z]],l=Limit[f[z],z->k]}
h[z_]:=Piecewise[{{1,z==k}},f[z]];
CartesianMap[h,{0,0.000001,0.2},{-4,4,0.2},Lines->10,
  PlotStyle->Hue[0.7],AspectRatio->1,
  Frame->True,Axes->False];
```

### 7.3 Sequences and Series

**Problem:** Find  $\lim_{n \rightarrow \infty} Z_n$ , where  $Z_n = (n^{1/4} + i(n^2 + 1))/n^2$ .

*Maple:*

```
Z_n:=(n^(1/4)+I*(n^2+1))/n^2; L1:=limit(Z_n, n=infinity);
```

*Mathematica:*

```
{zn=(n^(1/4)+I*(n^2+1))/n^2, l1=Limit[zn, n->Infinity]}
```

**Problem:** Show that the series  $\sum_{n=1}^{\infty} (1 + i(-1)^{n+1}n^2)/n^4$  converges.

*Maple:*

```
Z_n:=(1+I*(-1)^(n+1)*n^2)/n^4; ns:= 1..infinity;
S1:=sum(evalc(Re(Z_n)),n=ns)+I*sum(evalc(Im(Z_n)),n=ns);
S2:=convert(S1, StandardFunctions);
```

*Mathematica:*

```
z[n_]:=(1+I*(-1)^(n+1)*n^2)/n^4;
{l=z[n+1]/z[n]//Simplify,Limit[l,n->Infinity]}
```

**Problem:** Find the radius of convergence of the series  $f(z) = \sum_{n=0}^{\infty} \frac{z^{2n}}{2n!}$ .

*Maple:*

```
c:=n->1/(2*n)!; L:=simplify(c(n+1)/c(n)); A:=infinity;
R:=limit(1/L, n=A); sum(c(n)*z^(2*n), n=0..A);
```

*Mathematica:*

```
c[n_]:=1/(2*n)!; l=c[n+1]/c[n]//Simplify
r=Limit[1/l,n->Infinity]
Sum[c[n]*z^(2*n),{n,0,Infinity}]
```

*The Mandelbrot and Julia sets:* let  $f: \mathbb{C} \rightarrow \mathbb{C}$  be either the complex plane or the Riemann sphere. We consider  $f$  as a discrete dynamical system on the phase space  $\mathbb{C}$  and study the iteration behavior of  $f$ . Let the map  $z \mapsto z^2 + c$ , where  $z = x + iy$ ,  $c \in \mathbb{C}$ . We define the sequence  $\{z_n\}_{n=0}^{\infty}$  as follows:  $z_n = z_{n-1}^2 + c$ ,  $z_0 = 0$ .

*The Mandelbrot set*,  $M$ , generated for all values of  $c$ , consists of the values of  $c$  for which the sequence  $\{z_n\}_{n=0}^{\infty}$  does not diverge to infinity under iteration. For the Julia set  $J(f)$ , we consider a fix point  $c$  and generate the sequence  $\{z_n\}_{n=0}^{\infty}$  as follows:  $z_n = z_{n-1}^2 + c$ ,  $z_0 = z$ .

*The Julia set*,  $J$  consists of the values of  $z_0$  for which the sequence  $\{z_n\}_{n=0}^{\infty}$  does not diverge to infinity under iteration. The Mandelbrot and Julia sets produce fractals.

**Problem:** Construct the *Julia set* in 2D and 3D for the map  $z \mapsto z^3 - Cz$ ,  $C = 0.69 + 0.67i$ .

*Maple:*

```
with(plots): d:=3/2; CZ:=0.69+0.67*I;
Ops2D:=symbol=circle,style=point,color=red,axes=none;
Ops3D:=colorstyle=HUE,style=patchnogrid,
       axes=boxed,grid=[200,200];
JuliaSet2D := proc(C,t,d)
  local x,y,z,i,L,k,m: L:=[];
  z:=0: k:=200: m:=3:
  for x from -d*t to d*t do
    for y from -d*t to d*t do
      z:=x/t+I*y/t:
      for i from 0 while i<k and evalf(abs(z))<m do
        z:=z^3-C*z: od:
      if i=k then L:=[op(L),[x,y]]: fi: od: od:
  RETURN(L): end:
```

```
JuliaSet3D := proc(X,Y,C)
local Z,m,k,t; Z:=X+I*Y; k:=30; m:=3.0;
for t from 1 while t<k and evalf(abs(Z))<m do
    Z:=Z^3-C*Z; od; -t; end:
L_J:=JuliaSet2D(CZ,30,d): pointplot(L_J,0ps2D);
densityplot('JuliaSet3D'(x,y,CZ),x=-d..d,y=-d..d,0ps3D);
```

*Mathematica:*

```
{d=3/2,cz=0.69+0.67*I}
Set2DJulia[c_,t_,d_]:=Module[
{x,y,z=0,k=200,m=3.,l={},l1,i},
Do[Do[z=x/t+I*y/t;
i=0;While[i<k && N[Abs[z]]< m,z=z^3-c*z;i++];
If[i==k,l=Join[l,{x,y}],{y,-d*t,d*t},{x,-d*t,d*t}];
l1=Partition[l,2]];
Set3DJulia[x_,y_,c_]:=Module[{z,m=3.,k=30,t=1},z=x+I*y;
While[t<k && N[Abs[z]]< m,z=z^3-c*z;t++]; Return[-t]];
1J2D=Set2DJulia[cz,30,d];
ListPlot[1J2D,PlotStyle->{PointSize[0.03],Hue[0.7]},
AspectRatio->1];
pointsJ=Table[Set3DJulia[x,y,cz],{x,-d,d,0.01},
{y,-d,d,0.01}];
ListDensityPlot[pointsJ,Mesh->False,ColorFunction->Hue];
```

**Problem:** Construct the *Mandelbrot (Multibrot) set* in 3D for the map  $z \mapsto z^3$ .

*Maple:*

```
with(plots):
d:= 3/2; k:=200; m:=3; 0ps:=colorstyle=HUE,
style=patchnogrid, axes=boxed,grid=[200,200];
MandelbrotSet := proc(X,Y)
local Z,t; Z:=X+I*Y;
for t from 1 while t<k and evalf(abs(Z))<m do
    Z:=Z^3+(X+I*Y) od; -t; end:
densityplot('MandelbrotSet'(x,y),x=-d..d,y=-d..d,0ps);
```

*Mathematica:*

```
{d=3/2,k=200,m=3}
Set3DManelbrot[x_,y_]:=Module[{z,t=1},z=x+I*y;
  While[t<k && N[Abs[z]]< m,z=z^3+(x+I*y);t++];
  Return[-t]];
pointsM=Table[Set3DManelbrot[x,y],
 {x,-d,d,0.01},{y,-d,d,0.01}];
ListDensityPlot[pointsM,Mesh->False,ColorFunction->Hue];
```

## 7.4 Transformations and Mappings

**Problem:** The transformation  $w = z^2$  maps lines onto lines or parabolas. Find the image of the vertical line  $x = 5$ .

*Maple:*

```
z:=(x+I*y)^2; Eq1:={U=evalc(Re(z)),V=evalc(Im(z))};
Eq2:=subs(x=5, Eq1); Eq3:=eliminate(Eq2, y);
Sols:=[solve(Eq3[2][1], U)]; U:=V->expand(Sols[1]);
U(V); plot(U(V), V=-10..10);
```

*Mathematica:*

```
{z=(x+I*y)^2, eq1={u==ComplexExpand[Re[z]], 
 v==ComplexExpand[Im[z]]}, eq2=eq1/.{x->5}, 
 eq3=Eliminate[eq2,y]}
u[v_]:=Flatten[Solve[eq3,u]//Simplify][[1,2]]; u[v]
Plot[u[v],{v,-10,10},PlotStyle->{Blue,Thicknes[0.02]}];
```

*Conformal mapping*  $z = f(Z)$  on a rectangular region and the Riemann sphere.

*Maple:*

```
with(plots): f:=z-(5*z-1)/(5*z+1);
conformal(f(z),z=-1-2*I..1+2*I,-5-5*I..5+5*I,
           grid=[25,25], numxy=[140,140]);
conformal(cos(z)-sin(z), z=0-2*I..2*Pi+2*I, color=gold);
conformal3d(cos(z)-sin(z), z=0-2*I..2*Pi+2*I, color=white,
            grid=[20, 20], orientation=[17, 111]);
```

*Mathematica:*

```
<<Graphics`ComplexMap`;
f1[z_]:=(5*z-1)/(5*z+1); f2[z_]:=Cos[z]-Sin[z];
CartesianMap[f1,{ -1,1},{ -5,5},Lines->10,
    PlotStyle->Hue[0.7]];
PolarMap[f2,{0,1/4},{0,2*Pi,Pi/4},Lines->30,
    PlotStyle->Hue[0.9]];
CartesianMap[f2,{0,2},{0,2*Pi},PlotStyle->Hue[0.8]];
```

## 7.5 Differentiation and Integration

**Problem:** Find the derivatives of  $f_1(z) = z^2 + i5z - 1$  and  $f_2(z) = (f_1(z))^5$ .

*Maple:*

```
f1:=z->z^2+I*5*z-1; f2:=z->z^5; f3:=z->f2(f1(z));
diff(f1(z),z); diff(f3(z),z);
```

*Mathematica:*

```
f1[z_]:=z^2+I*5*z-1; f2[z_]:=z^5; f3[z_]:=f2[f1[z]];
{D[f1[z],z], D[f3[z],z]}
```

**Problem:** The Cauchy's integral theorem: let  $A$  be an open subset of  $B$  which is simply connected, let  $f : A \rightarrow B$  be a holomorphic function, and let  $C$  be a rectifiable path in  $A$  whose starting point is equal to its ending point. Then,  $\oint_C f(z) dz = 0$ . Evaluate this integral for  $f(z) = z^2$ .

*Maple:*

```
f:=z->z^2; C:=[r*exp(I*theta), theta=0..2*Pi];
Int(subs(z=C[1], f(z)*Diff(z,theta)),C[2])=
value(Int(subs(z=C[1], f(z)*Diff(z,theta)),C[2]));
```

*Mathematica:*

```
f[z_]:=z^2; z[theta_]:=r*Exp[I*theta];
Integrate[f[z[theta]]*z'[theta],{theta,0,2*Pi}]
```

**Problem:** Evaluate integrals around poles:  $\int_C f(z) dz = 2\pi i \sum_{k=1}^n \text{res}(f, a)$ .

For example,  $f(z) = \frac{5z - 2}{z^2 - z}$ .

*Maple:*

```
f:=(5*z-2)/(z^2-z);
R0:=residue(f,z=0); R1:=residue(f,z=1);
2*Pi*I*(R0+R1);
```

*Mathematica:*

```
f[z_]:=(5*z-2)/(z^2-z);
{r0=Residue[f[z],{z,0}], r1=Residue[f[z],{z,1}],
 2*Pi*I*(r0+r1)}
```

## 7.6 Singularities

*Singularities, series expansions about singularities, Laurent series.*

*Maple:*

```
f:=z->(z^2+a^2)^(-1/2); singular(f(z),z);
series(sqrt(z)+z^(-1/2),z=0,9);
series(cos(sqrt(z)),z=0,9);
series((1/sin(z))^2,z=0,9);
with(numapprox):
laurent(1/(x^3*sin(x^3)), x=0);
laurent(1/((x-1)*(x-3)),x=1);
laurent(1/((x-1)*(x-3)),x=3);
```

*Mathematica:*

```
f[z_]:=(z^2+a^2)^(-1/2); Solve[1/f[z]==0,z]
{Series[Sqrt[z]+z^(-1/2),{z,0,9}],
 Series[Cos[Sqrt[z]],{z,0,9}],
 Series[(1/Sin[z])^2,{z,0,9}]}
{Series[1/(x^3*Sin[x^3]),{x,0,9}],
 Series[1/((x-1)*(x-3)),{x,1,9}],
 Series[1/((x-1)*(x-3)),{x,3,9}]}
```

*Limits* at singularities.

*Maple*:

```
f1:=z->z^2; f2:=z->z^(1/2);
limit(f1(z),z=0,complex); limit(f2(z),z=0,complex);
f3:=z->z^(-2); f4:=z->z^(-1/2);
limit(f3(z),z=0,complex); limit(f4(z),z=0,complex);
f5:=z->exp(1/z); limit(f5(z),z=0,complex);
```

*Mathematica*:

```
f0[x_,y_]:=ComplexExpand[Conjugate[x+I*y]];
Limit[Limit[f0[x,y],x->x0],y->y0]
f1[z]:=z^2; f2[z]:=Sqrt[z];
{Limit[f1[z],z->0], Limit[f2[z],z->0]}
f3[z_]:=z^(-2); f4[z_]:=z^(-1/2); f5[z_]:=Exp[1/z];
{Limit[f3[z],z->0],Limit[f4[z],z->0],Limit[f5[z],z->0]}
```

# Chapter 8

## Special Functions

*Special functions* is a set of some classes of particular functions that have attractive or useful properties arising from solutions of theoretical and applied problems in different areas of mathematics.

*Special functions* can be defined by means of power and trigonometric series, series of orthogonal functions, infinite products, generating and distribution functions, integral representations, sequential differentiation, transcendental, differential, difference, integral, and functional equations.

*Maple* includes over 200 special functions. We will consider some the most important special functions (see `?inifcn`, `?index[package]`, `?index[function]`, `?FunctionAdvisor`).

*Mathematica* includes all the common special functions of mathematical physics found in standard handbooks (see `MasterIndex`, `Special Functions`, `MathematicalFunctions`). It should be noted that the definitions (including normalizations and special values) of any particular special function can be different in handbooks and also in *Maple* and *Mathematica* (e.g., the Mathieu functions). We will discuss some the most important special functions.

### 8.1 Special Integrals

*Gamma*, *Beta*, *digamma*, and *polygamma* functions.

*Maple*: `GAMMA`, `Beta`, `Psi(x)`, `Psi(n,x)`.

```
plot(GAMMA(x),x=-5..5,-20..20,numpoints=500,color=blue);
```

```
GAMMA(z+2); convert(GAMMA(z+2), factorial);
convert(z!, GAMMA); simplify(z*GAMMA(z)); GAMMA(1/2);
GAMMA(-1/2); GAMMA(5/2); Beta(2,9); expand(Psi(73));
evalf(expand(Psi(2,73))); evalf(Psi(2,73));
```

*Mathematica:* Gamma, Beta, PolyGamma[x], PolyGamma[n,x].

```
Plot[Gamma[x], {x, -5, 5}, PlotRange -> {-20, 20},
    PlotStyle -> Hue[0.9]];
{z*Gamma[z] // FunctionExpand, Gamma[1 - 2*I] // N,
 D[Gamma[z], z], Gamma[1/2], Gamma[-1/2], Gamma[5/2]}
{Beta[2, 9], N[{Erf[1], Erfc[1]}]}
{PolyGamma[73], PolyGamma[2, 73] // FunctionExpand,
 PolyGamma[2, 73] // N}
```

*Exponential, logarithmic, polylogarithmic, and trigonometric integrals.*

*Maple:* Ei, Li, dilog, Si, Ci.

```
expand(Ei(5, x)); evalf(Si(5)); convert(Li(x), Ei);
convert(Ci(x), Ei); dilog(-1/2);
```

*Mathematica:* ExpIntegralE, LogIntegral, PolyLog,  
SinIntegral, CosIntegral.

```
{ExpIntegralE[5, x] // FunctionExpand, SinIntegral[5] // N}
Series[LogIntegral[x], {x, 2, 9}] // Normal // N
{PolyLog[2, -1/2] // N, D[CosIntegral[x], x],
 Limit[Integrate[SinIntegral[x], x], x -> 0]}
```

*Error functions, Fresnel integrals.*

*Maple:* erf, erfc, FresnelC, FresnelS.

```
evalf(erf(1)); evalf(erfc(1));
plot([FresnelC(z), FresnelS(z), z = -10..10], -1..1, -1..1,
    scaling = constrained, color = blue, axes = boxed);
```

*Mathematica:* Erf, Erfc, FresnelC, FresnelS.

```
{Erf[1]/N, Erfc[1]/N}
ParametricPlot[{FresnelC[z], FresnelS[z]}, {z, -10, 10},
PlotRange -> {{-1, 1}, {-1, 1}}, AspectRatio -> 1,
PlotStyle -> Hue[0.7], Frame -> True];
```

*Elliptic integrals* and functions.

*Maple:* EllipticF, EllipticK, EllipticE, EllipticPi.

```
EllipticF(0.1,0.2); EllipticK(0.1); EllipticE(0.1,0.2);
EllipticE(0.1);EllipticPi(0.1,0.1,0.2);EllipticPi(0.1,0.2);
z:=1/2*sqrt(10+25*I); k:=1/2*sqrt(2)-1/2*I*sqrt(2);
evalf(EllipticE(z, k)); z1:=evalf(z); k1:=evalf(k);
int(sqrt(evalc(1-k1^2*t^2))/sqrt(evalc(1-t^2)),t=0..z1);
```

*Mathematica:* EllipticF, EllipticK, EllipticE, EllipticPi.

```
{EllipticF[0.1,0.2], EllipticK[0.1], EllipticE[0.1,0.2]
EllipticE[0.1],EllipticPi[0.1,0.1,0.2],EllipticPi[0.1,0.2]}
{z=1/2*Sqrt[10+25*I], k=1/2*Sqrt[2]-1/2*I*Sqrt[2],
EllipticE[z,k]/N, z1=N[z], k1=N[k],
NIntegrate[Sqrt[1-k1^2*t^2]/Sqrt[1-t^2],{t,0,z1}]}
```

## 8.2 Orthogonal Polynomials

*Classical orthogonal polynomials* and the corresponding functions, e.g., Gegenbauer, Hermite, Jacobi, Laguerre, Legendre, Chebyshev polynomials.

*Maple:*

```
with(orthopoly): k:=7; m:=x=1;
P_Gegenbauer:=[G(n,2,x)$n=0..k]; subs(m,P_Gegenbauer);
P_Hermite:=[H(n,x) $ n=0..k]; subs(m,P_Hermite);
P_Jacobi:=[P(n,2,4,x) $ n=0..k]; subs(m,P_Jacobi);
P_Laguerre:=[L(n,x) $ n=0..k]; subs(m,P_Laguerre);
P_Legendre:=[P(n,x)$n=0..k]; subs(m,P_Legendre);
P_ChebyshevI:=[T(n,x)$n=0..k]; subs(m,P_ChebyshevI);
plot({T(n,x)$n=0..k},x=-1..1,color=blue,axes=boxed);
P_ChebyshevII:=[U(n,x)$n=0..k]; subs(m,P_ChebyshevII);
F := HermiteH(4,x); simplify(F, 'HermiteH');
```

*Mathematica:*

```
{k=7, m={x->1}}
pGegenbauer=Table[GegenbauerC[n,2,x],{n,0,k}]
pHermiteH=Table[HermiteH[n,x],{n,0,k}]
pJacobiP=Table[JacobiP[n,2,4,x],{n,0,k}]//Simplify
pLaguerreL=Table[LaguerreL[n,x],{n,0,k}]
pLegendreP=Table[LegendreP[n,x],{n,0,k}]
pChebyshevT=Table[ChebyshevT[n,x],{n,0,k}]
Plot[Evaluate[pChebyshevT],{x,-1,1},
      PlotStyle->Blue,Frame->True];
pChebyshevU=Table[ChebyshevU[n,x],{n,0,k}]
{pGegenbauer/.m,pHermiteH/.m,pJacobiP/.m,
 pLaguerreL/.m,pLegendreP/.m,pChebyshevT/.m,
 pChebyshevU/.m}
```

### 8.3 Trascendental Equations

*Special functions* defined by trascendental equations.

**Problem:** The Lambert  $W$  function, the two real-valued branches of  $W$ .

*Maple:* The Lambert  $W$  function, *LambertW*.

```
with(plots): alias(W=LambertW): solve(y*exp(y)-z,y);
diff(W(z),z); series(W(z),z,9); W(-1.); A:=thickness=2;
G1:=plot(W(x)^(-1),x=-0.3678..0,-5..1, A, linestyle=2);
G2:=plot(W(x),x=-0.3678..1,-5..1, A): display([G1,G2]);
```

*Mathematica:* The Lambert  $W$  function, *ProductLog*.

```
Reduce[y*Exp[y]-z==0,y]
{D[ProductLog[z],z],Series[ProductLog[z],{z,0,9}],
 ProductLog[-1.]}
g1=Plot[ProductLog[x]^(-1),{x,-0.3678,0},PlotRange->
 {{-0.3678,0},{-5.,1.}},PlotStyle->{Blue,Dashing[{0.01}]},
 DisplayFunction->Identity];
g2=Plot[ProductLog[x],{x,-0.3678,1},PlotRange->
 {{-0.3678,1},{-5.,1.}},PlotStyle->{Blue,Thicknes[0.02]},
 DisplayFunction->Identity];
Show[{g1,g2},DisplayFunction->$DisplayFunction,
 AspectRatio->1,Axes->False];
```

## 8.4 Differential Equations

*Special functions* defined by differential equations, the Bessel functions, the modified Bessel functions, the Hankel functions.

*Maple:*

```
with(plots): C:=color=[green,blue,magenta];
S:=scaling=constrained; B:=axes=boxed;
plot([BesselJ(n,x) $ n=0..2],x=0..10,C,B);
plot([BesselY(n,x) $ n=0..2],x=0..10,-1..1,C,B);
plot([BesselI(n,x) $ n=1..3],x=-Pi..Pi,C,B);
plot([BesselK(n,x) $ n=0..2],x=0..Pi,-10..10,C,B);
conformal(HankelH1(0,x),x=1+I..-1+2*I,color=blue,S,B);
conformal(HankelH2(0,x),x=1-I..-1-2*I,color=blue,S,B);
asympt(BesselJ(0,x),x,1);convert(HankelH2(n,x),Bessel);
evalf(BesselJZeros(1,1..9));evalf(BesselYZeros(1,1..9));
```

*Mathematica:*

```
SetOptions[Plot,PlotStyle->{Green,Blue,Magenta},
           AspectRatio->1,Frame->True];
Plot[Evaluate[Table[BesselJ[n,x],{n,0,2}]],{x,0,10}];
Plot[Evaluate[Table[BesselY[n,x],{n,0,2}]],{x,0,10},
     PlotRange->{{0,10},{-1,1}}];
Plot[Evaluate[Table[BesselI[n,x],{n,1,3}]],{x,-Pi,Pi},
     PlotRange->{{-Pi,Pi},{-4,4}}];
Plot[Evaluate[Table[BesselK[n,x],{n,0,2}]],{x,0,Pi},
     PlotRange->{{0,Pi},{-10,10}}];
<<Graphics`ComplexMap`;
trHankelH1[n_,x_]:=BesselJ[n,x]+I*BesselY[n,x];
trHankelH2[n_,x_]:=BesselJ[n,x]-I*BesselY[n,x];
CartesianMap[trHankelH1[0,#]&,{ -1,1},{2,1},Lines->10,
             PlotStyle->Hue[0.7],Frame->True];
CartesianMap[trHankelH2[0,#]&,{ -1,1},{-2,-1},Lines->10,
             PlotStyle->Hue[0.7],Frame->True];
<<NumericalMath`BesselZeros`;
{BesselJZeros[1,9], BesselYZeros[1,{9,17}],
 BesselYZeros[1,9,WorkingPrecision->30],
 BesselJZerosInterval[1,{9,17}]}
```

*The Mathieu* functions.

*Maple*:

```
Ops:=scaling=constrained,axes=boxed;
plot(MathieuCE(0,10,x), x=-Pi..Pi, Ops);
plot(MathieuSE(1,10,x), x=-Pi..Pi, Ops);
plot(MathieuCE(10,20,x),x=-Pi..Pi, Ops);
MathieuCE(n,0,x); MathieuSE(n,0,x);
series(MathieuA(2,q),q,10); evalf(MathieuB(2,20));
MathieuFloquet(a,0,x); MathieuExponent(a,0);
MathieuC(a,0,x); MathieuS(a,0,x); MathieuCEPrime(n,0,x);
```

*Mathematica*:

```
SetOptions[Plot,PlotStyle->Blue,AspectRatio->1,
           Frame->True];
Plot[Re[MathieuC[MathieuCharacteristicA[0,20],20,x]],
     {x,-Pi,Pi}];
Plot[Re[MathieuS[MathieuCharacteristicB[1,10],10,x]],
     {x,-Pi,Pi}];
Plot[Re[MathieuC[MathieuCharacteristicA[10,20],20,x]],
     {x,-Pi,Pi}];
MathieuC[MathieuCharacteristicA[n,0],0,x]//FunctionExpand
Assuming[n>0,MathieuS[MathieuCharacteristicB[n,0],0,x]
         //FunctionExpand//Simplify]
N[MathieuCharacteristicB[2,20],10]
MathieuCharacteristicExponent[a,0]
{MathieuC[a,0,x], MathieuS[a,0,x], MathieuCPrime[n,0,x]}
{q=20,a1=0,an=50}
SetOptions[ListPlot,Axes->False,Frame->True,
           AspectRatio->1];
g1=ListPlot[Table[{a,MathieuCharacteristicA[a,q]},
                  {a,a1,an}],PlotStyle->{Hue[0.7],PointSize[0.02]},
            DisplayFunction->Identity];
g2=ListPlot[Table[MathieuCharacteristicB[a,q],
                  {a,a1+3,an}],PlotStyle->{Hue[0.9],PointSize[0.02]},
            DisplayFunction->Identity];
Show[{g1,g2},DisplayFunction->$DisplayFunction];
```

*The Legendre functions* and the associated Legendre functions.

*Maple:*

```
F_LP := LegendreP(2,x); simplify(F_LP,'LegendreP');
F_LQ := LegendreQ(2,x); simplify(F_LQ,'LegendreQ');
F_LPA := LegendreP(2,2,x); simplify(F_LPA,'LegendreP');
F_LQA := LegendreQ(2,2,x); simplify(F_LQA,'LegendreQ');
convert(LegendreP(2,a,x),hypergeom);
```

*Mathematica:*

```
eqLeg=(1-x^2)*y''[x]-2*x*y'[x]+n*(n-1);
{sol=DSolve[eqLeg==0,y[x],x],
test1=Simplify[eqLeg/.sol][[1]],test1==Simplify[eqLeg]}
Map[FunctionExpand,{LegendreP[2,x],LegendreQ[2,x],
LegendreP[2,2,x],LegendreQ[2,2,x],LegendreQ[2,2,2,x]}]
```

## 8.5 Infinite Series

*Special functions* defined by infinite series, the Riemann Zeta function, hypergeometric functions.

*Maple:*

```
sum(1/i^9,i=1..infinity)=Zeta(9.);
plot(Zeta(x),x=-1..2,-10..10); map(simplify,
[z*hypergeom([1,1],[2],-z), hypergeom([-3],[],x),
hypergeom([-n,beta],[beta],-z),x*hypergeom([2,2],[1],x),
hypergeom([1/2],[3/2],-x^2),hypergeom([],[],-(x/2)^2)]);
convert(hypergeom([1,2],[3/2],z), StandardFunctions);
```

*Mathematica:*

```
{ex1=Sum[1/i^9,{i,1,Infinity}],N[ex1],ex1==Zeta[9]}
Plot[Zeta[x],{x,-1,2},PlotRange->{{-1,2},{-10,10}},
PlotStyle->Blue,AspectRatio->1];
{z*HypergeometricPFQ[{1,1},{2},-z],
HypergeometricPFQ[{-3},[],x],
HypergeometricPFQ[{-n,beta},{beta},-z]}
```

```

{x*HypergeometricPFQ[{2,2},{1},x],
 HypergeometricPFQ[{1/2},{3/2},-x^2],
 HypergeometricPFQ[{}, {1}, -(x/2)^2}]
Series[HypergeometricOF1[a,x],{x,0,10}]
Integrate[Hypergeometric1F1[a,b,x],x]
Hypergeometric2F1[2,2,2,z]
D[HypergeometricU[a,b,x],{x,2}]

```

## 8.6 Distribution Functions

The Dirac  $\delta$ -function, the Heaviside unit step function.

*Maple:*

```

delta:=t->Dirac(t); H:=t->Heaviside(t);
Int(delta(t), t)=int(delta(t),t);
Int(H(t),t)=int(H(t),t); Diff(H(t),t)=diff(H(t),t);
Diff(delta(t),t)=diff(delta(t),t);
Diff(delta(t),t$2)=diff(delta(t),t$2);
Int(delta(t-a)*f(t), t=-infinity..infinity)
    =int(delta(t-a)*f(t), t=-infinity..infinity);
convert(Heaviside(t-1)+Heaviside(t-2),piecewise);
assume(b: simplify(convert(piecewise(t>=0 and t<b,0,
t>=b and t<a,1,t>=a,f(t)),Heaviside));

```

*Mathematica:*

```

delta[t_]:=DiracDelta[t]; H[t_]:=UnitStep[t];
{Integrate[delta[t],t],Integrate[H[t],t]}
{H'[t],delta'[t],delta''[t]}
Integrate[delta[t-a]*f[t],{t,-Infinity,Infinity}]
Assuming[a\[Element] Reals,
 Integrate[delta[t-a]*f[t],{t,-Infinity,Infinity}]]
{f1=H[(t-1)*(t-2)]//FunctionExpand,
 PiecewiseExpand[f1],TraditionalForm[f1]}
Plot[UnitStep[Cos[x]],{x,-20,20},Frame->True,
 PlotStyle->Blue,Axes->False];
Plot[Evaluate[Sqrt[i/Pi]*Exp[-i*x^2]/.{i->{10,50}}],
 {x,-Pi,Pi},Frame->True,PlotStyle->Blue,Axes->False,
 PlotRange->All];

```

# Chapter 9

## Integral and Discrete Transforms

### 9.1 Laplace Transforms

In *Maple*, the integral transforms (e.g., Fourier, Hilbert, Laplace, Mellin integral transforms) can be studied with the aid of the package `inttrans`.

In *Mathematica*, the integral Laplace transforms are defined by the two functions `LaplaceTransform`, `InverseLaplaceTransform`.

**Problem:** Find the integral Laplace transforms for different functions.

*Maple*:

```
with(inttrans); with(plots); assume(a>0);
f1:=t->piecewise(t>=0 and t<=1,-1,t>1,1);
f2:=t->2*Heaviside(t-1)-Heaviside(t);
G:=array(1..2);
G[1]:=plot(f1(t),t=0..10,color=blue):
G[2]:=plot(f2(t),t=0..10,color=magenta):
display(G, scaling=constrained);
map(simplify,
     [laplace(f1(t),t,s),laplace(f2(t),t,s)]);
laplace(diff(u(x),x$2),x,s);
laplace(Dirac(t-a),t,s);
simplify(laplace(exp(a*t)*erf(sqrt(a*t)),t,s));
simplify(laplace(F(t-a)*Heaviside(t-a),t,s));
```

*Mathematica:*

```
f1[t_]:=Piecewise[{{{-1,0<=t<=1},{1,t>1}}];
f2[t_]:=2*UnitStep[t-1]-UnitStep[t];
{f1[t], f2[t]}
g1=Plot[f1[t],{t,0,10},PlotStyle->{Hue[0.7],
    Thickness[0.01]},DisplayFunction->Identity];
g2=Plot[f2[t],{t,0,10},PlotStyle->{Hue[0.9],
    Thickness[0.01]},DisplayFunction->Identity];
Show[GraphicsArray[{g1,g2}],
    DisplayFunction->$DisplayFunction];
Map[FullSimplify,
{LaplaceTransform[f1[t],t,s],
 FullSimplify[LaplaceTransform[f1[x],x,s]]===
 FullSimplify[LaplaceTransform[f2[x],x,s]],
 LaplaceTransform[D[u[x],{x,2}],x,s],
 Assuming[a>0,LaplaceTransform[DiracDelta[t-a],t,s]],
 Assuming[a>0,LaplaceTransform[Exp[a*t]*Erf[Sqrt[a*t]],t,s]],
 Assuming[a>0,LaplaceTransform[F[t-a]*UnitStep[t-a],t,s]],
 LaplaceTransform[t^2*Cos[2*t],t,s]}]
```

**Problem:** Find the inverse Laplace transforms for different functions.

*Maple:*

```
with(inttrans);
f1:=t->2*Heaviside(t-1)-1;
F1:=unapply(laplace(f1(s),s,t),t);
invlaplace(F1(s),s,t);
invlaplace(1/s^3,s,t); invlaplace(1/(s-a),s,t);
assume(a>0,x>0); simplify(invlaplace(exp(-s*x),s,t));
simplify(invlaplace((s^2+a^2)^(-1/2),s,t));
```

*Mathematica:*

```
f1[t_]:=2*UnitStep[t-1]-1; f1[t]
F1[T_]:=LaplaceTransform[f1[t],t,s]/.{t->T}
InverseLaplaceTransform[F1[s],s,t]===(f1[t]//FullSimplify)
{InverseLaplaceTransform[1/s^3,s,t],
 InverseLaplaceTransform[1/(s-a),s,t],
 Assuming[a>0 && x>0,
     InverseLaplaceTransform[Exp[-s*x],s,t]//Simplify],
 Assuming[a>0,
     InverseLaplaceTransform[1/(Sqrt[s]+a),s,t]//Simplify]}
```

*Applications* to ordinary differential equations.

**Problem:** Obtain the solution of the initial value problem

$$y' + ay = e^{-at}, \quad y(0) = 1.$$

*Maple:*

```
with(inttrans);
ODE:=diff(y(t),t)+a*y(t)=exp(-a*t);
Eq1:=laplace(ODE,t,p); Eq2:=subs(y(0)=1,Eq1);
Eq3:=solve(Eq2,laplace(y(t),t,p));
Sol:=invlaplace(Eq3,p,t);
dsolve({ODE,y(0)=1},y(t)); Sol1:=subs(a=7,Sol);
plot(Sol1,t=0..1,color=blue);
```

*Mathematica:*

```
ode={y'[t]+a*y[t]==Exp[-a*t]}
eq1=LaplaceTransform[ode,t,s]/.{y[0]->1}
eq2=Solve[eq1,LaplaceTransform[y[t],t,s]]
sol=Map[InverseLaplaceTransform[#,s,t]&,eq2,{3}]/.{a->7}
DSolve[{ode,y[0]==1},y[t],t]
Plot[y[t]/.sol,{t,0,1},
 PlotStyle->{Hue[0.7],Thickness[0.01]}];
```

*Applications* to partial differential equations.

**Problem:** Obtain the solution of the initial boundary-value problem for the wave equation describing the transverse vibration of a semi-infinite string,

$$u_{tt} = c^2 u_{xx}, \quad 0 \leq x < \infty, \quad t > 0,$$

with the initial and boundary conditions

$$u(x, 0) = 0, \quad u_t(x, 0) = 0, \quad u(0, t) = Af(t), \quad u(x, t) \rightarrow 0, \text{ as } x \rightarrow 0.$$

*Maple:*

```
with(inttrans): with(PDEtools):
assume(c>0,s>0);
declare(u(x,t),U(x)); ON;
Eq1:=diff(u(x,t),t$2)-c^2*diff(u(x,t),x$2);
Eq2:=laplace(Eq1,t,s);
Eq3:=subs({laplace(u(x,t),t,s)=U(x),
           laplace[x,x]=diff(U(x),x$2)},Eq2);
IC1:={u(x,0)=0,D[2](u)(x,0)=0};
IC2:=laplace(IC1,t,s);
BC1:=u(0,t)=A*f(t); BC2:=laplace(BC1,t,s);
BC3:=subs({laplace(u(0,t),t,s)=U(0),
            laplace(f(t),t,s)=F(s)},BC2);
Eq4:=subs(IC1,Eq3);
Sol:=dsolve({Eq4} union {BC3},U(x));
C2:=evala(solve(limit(rhs(Sol),x=infinity)=0,_C2));
U1:=rhs(subs({_C2=C2,F(s)=laplace(f(t),t,s)},Sol));
assume(x>0); Sol2:= simplify(invlaplace(U1,s,t));
Sol_F:=convert(Sol2,piecewise,t);
```

*Mathematica:*

```
eq1=D[u[x,t],{t,2}]==c^2*D[u[x,t],{x,2}]
eq2=LaplaceTransform[eq1,t,s]/
  {LaplaceTransform[u[x,t],t,s]->u1[x],
   LaplaceTransform[D[u[x,t],{x,2}],t,s]->u1''[x]}
ic1={u[x,0]->0,(D[u[x,t],t]/.{t->0})->0}
ic2=Map[LaplaceTransform[#,t,s]&,ic1,{2}]
eq3=(eq2/.ic1)
bc1=u[0,t]==a*f[t]
bc2=LaplaceTransform[bc1,t,s]/
  {LaplaceTransform[u[0,t],t,s]->u1[0],
   LaplaceTransform[f[t],t,s]->f1[s]}
{sol=DSolve[eq3,u1[x],x], sol1=u1[x]/.sol}
l1=Limit[sol1[[1,1]],x->Infinity,
          Assumptions->{c>0,s>0}]//Simplify
l2=Limit[sol1[[1,2]],x->Infinity,
          Assumptions->{c>0,s>0}]//Simplify
{sol2=sol1/.{C[1]->0}, sol3=sol2/.{C[2]->bc2[[2]]}}
u2=sol3/.f1[s]->LaplaceTransform[f[t],t,s]
solFin=InverseLaplaceTransform[u2,s,t]//PiecewiseExpand
```

*Distributions* and Laplace transform.

*Maple:*

```
with(inttrans); assume(a>0,b>0);
simplify(laplace(a*(t-b)^2*Heaviside(t-b),t,p));
factor(invlaplace(exp(-a*p)/p^2, p, t));
```

*Mathematica:*

```
Assuming[{a>0,b>0}, FullSimplify[
 LaplaceTransform[a*(t-b)^2*UnitStep[t-b],t,s]]
Assuming[a>0,
 InverseLaplaceTransform[Exp[-a*s]/s^2,s,t]]
```

**Problem:** Find the solution of the inhomogeneous Cauchy problem

$$Ay'' + By = f(t), \quad y(0) = 0, \quad y'(0) = 0,$$

where  $f(t)$  is a given function representing a source term

$$f(t) = H(t) - H(t - \pi).$$

*Maple:*

```
with(inttrans);
f:=t->Heaviside(t)-Heaviside(t-Pi);
ODE:=A*diff(y(t),t$2)+B*y(t)=f(t);
Eq1:=laplace(ODE,t,s);
Eq2:=subs({y(0)=0,D(y)(0)=0},Eq1);
Eq3:=simplify(solve(Eq2,laplace(y(t),t,s)));
Sol:=invlaplace(Eq3,s,t);
combine(dsolve({ODE,y(0)=0,D(y)(0)=0},y(t)));
convert(Sol,piecewise);
Sol1:=eval(Sol,{A=5,B=10});
plot(Sol1, t=0..10*Pi);
```

*Mathematica:*

```
f[t_]:=UnitStep[t]-UnitStep[t-Pi];
ode={a*y''[t]+b*y[t]==f[t]};
eq1=LaplaceTransform[ode,t,s]/.{y[0]->0,y'[0]->0}
eq2=Solve[eq1,LaplaceTransform[y[t],t,s]]
eq3=eq2[[1,1,2]]
sol=InverseLaplaceTransform[eq3,s,t]
DSolve[{ode,y[0]==0,y'[0]==0},y[t],t]//Simplify
{sol//FullSimplify, f=sol/.{a->5,b->10}}
Plot[f,{t,0,10*Pi},PlotStyle->{Hue[0.8],Thickness[0.01]}];
```

*Initial value problems* for ODE systems with the Laplace transform.

**Problem:** Applying the Laplace transform, solve the initial value problem

$$x' - 2y = t, \quad 4x + y' = 0, \quad x(0) = 1, \quad y(0) = 0,$$

and graph the solution.

*Maple:*

```
with(inttrans):
ICs:={x(0)=1,y(0)=0};
OdeSys:={D(x)(t)-2*y(t)=t,4*x(t)+D(y)(t)=0};
Sol1:=dsolve(OdeSys union ICs,{x(t),y(t)});
Sol2:=dsolve(OdeSys union ICs,{x(t),y(t)},method=laplace);
odetest(Sol1, OdeSys union ICs);
Eq1:=laplace(OdeSys,t,s); Eq2:=subs(ICs,Eq1);
Eq3:=solve(Eq2,{laplace(x(t),t,s),laplace(y(t),t,s)});
Sol3:=invlaplace(Eq3,s,t); assign(Sol3):
plot([x(t),y(t),t=0..Pi],color=blue,thickness=3);
```

*Mathematica:*

```
odeSys={x'[t]-2*y[t]==t,4*x[t]+y'[t]==0}
eq1=LaplaceTransform[odeSys,t,s]
eq2=Solve[eq1,{LaplaceTransform[x[t],t,s],
LaplaceTransform[y[t],t,s]}]
sol1=Map[InverseLaplaceTransform
          #[#,s,t]&,eq2,{3}]/.{x[0]->1,y[0]->0}
sol2=DSolve[{odeSys,x[0]==1,y[0]==0},
{x[t],y[t]},t]//Simplify
ParametricPlot[Evaluate[{x[t],y[t]}/.sol1],
{t,0,Pi},PlotStyle->{Hue[0.5],Thickness[0.01]}];
```

## 9.2 Integral Fourier Transforms

In *Maple*, integral Fourier transforms are defined with the functions `fourier`, `invfourier`, `fouriercos`, `fouriersin`. It should be noted that in *Maple* the product of normalization factors is equal to  $1/(2\pi)$ .

In *Mathematica*, integral Fourier transforms are defined with a family of functions `FourierTransform`, `InverseFourierTransform`, etc. (for more details, see `?*Fourier*`). Note that there is the option `FourierParameters` allowing one to choose different conventions used for defining Fourier transforms.

**Problem:** Find the Fourier transform for  $f(t) = e^{-t^2}$ ,  $f(t) = e^{-a(t-b)^2}$ .

*Maple*:

```
with(inttrans);
f1:=t->exp(-t^2); f1(t);
F1:=k->fourier(f1(t),t,k); F1(k);
plot({f1(x),F1(x)}, x=-Pi..Pi, color=[blue, green]);
assume(a>0,b>0); f2:=t->exp(-a*(t-b)^2);
F2:=k->fourier(f2(t),t,k); f2(t); factor(F2(k));
```

*Mathematica*:

```
f1[t_]:=Exp[-t^2]; f2[t_]:=Exp[-a*(t-b)^2];
g[s_]:=FourierTransform[f1[t],t,s]
g1[s_]:=FourierTransform[f1[t],t,s,
    FourierParameters->\{1,-1\}]
{f1[t], g[s], g1[s]}
Plot[Evaluate[{f1[x],g1[x]}],{x,-10,10},PlotRange->
{{{-10,10},{0,2}},PlotStyle->\{Hue[0.5],Hue[0.7]\},
 AspectRatio->1];
g2[s_]:=Assuming[a>0&&b>0,FourierTransform[f2[t],t,s,
    FourierParameters->\{1,-1\}]]
{f2[t],Evaluate[g2[s]]}]
```

**Problem:** Find the Fourier transform for  $f(t) = \begin{cases} 1, & |t| < a \\ 0, & |t| \geq a. \end{cases}$

*Maple:*

```
with(inttrans);
assume(a>0); C:=color=[red,blue];
f1:=piecewise(abs(t)<a,1,0);
f2:=unapply(convert(f1,Heaviside),t);
F1:=k->fourier(f2(t),t,k); f2(t); F1(k); limit(F1(k),k=0);
L:= [f2(x),F1(x)]; L1:=subs(a=1,L[1..2]);
plot(L1,x=-3*Pi..3*Pi,thickness=[3,2],C);
```

*Mathematica:*

```
f1[t_]:=Piecewise[{{1,Abs[t]<a},{0,Abs[t]>=a}}];
g1[s_]:=Assuming[a>0,FourierTransform[f1[t],t,s,
    FourierParameters->{1,-1}]];
{f1[t], g1[s], Limit[g1[s],s->0,Assumptions->{a>0}]}
{l={f1[x],g1[x]}, l1=l/.{a->1}}
Plot[Evaluate[l1],{x,-3*Pi,3*Pi},PlotStyle->
 {{Red,Thickness[0.01]},{Blue,Thickness[0.01]}]];
```

**Problem:** Find the inverse Fourier transform for  $1/p^2$ , the Fourier Sine transform for the exponential integral  $Ei(x)$ , and the Fourier Cosine transform for the Bessel function  $BesselY(0,x)$ .

*Maple:*

```
with(inttrans);
simplify(invfourier(1/p^2,p,t));
fouriersin(Ei(x),x,p); fouriercos(BesselY(0,x),x,p);
```

*Mathematica:*

```
{InverseFourierTransform[1/p^2,p,t],
FourierSinTransform[ExpIntegralEi[x],x,p],
FourierCosTransform[BesselY[0,x],x,p]}
```

*Applications* to partial differential equations.

**Problem:** Obtain the d'Alembert solution of the initial-value problem for the wave equation

$$u_{tt} = c^2 u_{xx}, \quad -\infty < x < \infty, \quad t > 0,$$

with the initial conditions  $u(x,0) = f(x)$ ,  $u_t(x,0) = g(x)$ .

*Maple:*

```
with(inttrans):with(PDEtools):
declare(u(x,t),U(t)); ON;
Eq1:=diff(u(x,t),t$2)-c^2*diff(u(x,t),x$2);
Eq2:=subs(fourier(u(x,t),x,k)=U(t),fourier(Eq1,x,k));
IC1:={u(x,0)=f(x),D[2](u)(x,0)=g(x)};
IC2:=fourier(IC1,x,k);
IC3:=subs({fourier(D[2](u)(x,0),x,k)=D(U)(0),
           fourier(u(x,0),x,k)=U(0),fourier(f(x),x,k)=F(k),
           fourier(g(x),x,k)=G(k)}, IC2);
sys:={Eq2} union IC3;
Sol:=dsolve(sys,U(t));
U1:=convert(rhs(Sol),exp);
Sol2:=invfourier(U1,k,x);
Sol3:=factor(combine(convert(Sol2*(2*Pi),int)));
N1:=factor(subs([G(k)=0],op(1,Sol3)));
N2:=subsop(2=1, 3=f(x+c*t)+f(x-c*t), N1);
N3:=factor(subs([F(k)=0],op(1,Sol3)));
N4:=simplify(N3/G(k)/int(exp(I*k*x),xi=x-c*t..x+c*t));
u_Dalembert:=N2+N4*int(g(xi),xi=x-c*t..x+c*t);
```

*Mathematica:*

```
eq1=D[u[x,t],{t,2}]==c^2*D[u[x,t],{x,2}]
eq2=Map[FourierTransform[#,x,k]&,eq1]/
  {FourierTransform[u[x,t],x,k]->U[t],
   FourierTransform[D[u[x,t],{t,2}],x,k]->U'[t]}
ic1={u[x,0]==f[x],(D[u[x,t],t]/.{t->0})==g[x]}
ic2=Map[FourierTransform[#,x,k]&,ic1,{2}]
ic3=ic2/
  {FourierTransform[(D[u[x,t],t]/.{t->0}),x,k]->U'[0],
   FourierTransform[u[x,0],x,k]->U[0],
   FourierTransform[f[x],x,k]->f1[k],
   FourierTransform[g[x],x,k]->g1[k]}
{sys=Union[{eq2},ic3], sol=DSolve[sys,U[t],t],
 sol1=U[t]/.sol//Simplify,
 u1=sol1/.{Sin[a_]:>-I/2*(Exp[I*a]-Exp[-I*a]),
            Cos[a_]:>1/2*(Exp[I*a]+Exp[-I*a])}
           //Expand,
 u2=Integrate[u1*Exp[x*k*I],{k,-Infinity,Infinity}]
           //Simplify}
```

```

{n1=u2[[1,1]]/.g1[k]->0//ExpandAll,
n11=Collect[n1,f1[k]/2],
n2=n11/.{f1[k]->1,n11[[2]]->f[x+c*t]+f[x-c*t]},
n3=u2[[1,1]]/.f1[k]->0//ExpandAll,
n4=(n3/g1[k])/Integrate[Exp[I*k*\[Xi]],
{\[Xi],x-c*t,x+c*t}]/Simplify}
uDalembert=n2+n4*Integrate[g1[\[Xi]],{\[Xi],x-c*t,x+c*t}]

```

### 9.3 Discrete Fourier Transforms

In *Maple*, with the new package `DiscreteTransforms` (*Maple*  $\geq 9$ ), the *fast Fourier transform and inverse transform* of a single or multi-dimensional data can be calculated. It should be noted that these functions work in hardware precision (`Digits:=15`), but very fast (in compiled code). Also *fast Fourier transforms* can be calculated with the functions `FFT` and `iFFT`.

In *Mathematica*, the *discrete Fourier transforms* are defined with the functions `Fourier` and `InverseFourier`. It should be noted that there is the option `FourierParameters` allowing one to choose different conventions used for defining discrete Fourier transforms. In the algorithm known as the Fast Fourier Transform (FFT), it is recommended that the length of the original list be a power of 2. This can reduce the computation time of discrete Fourier transform. But *Mathematica* can work well with lists whose lengths are different from a power of 2.

**Problem:** Generate a signal, construct the corresponding frequency spectrum, and recover the original signal.

*Maple*:

```

with(DiscreteTransforms): with(plots):
with(LinearAlgebra): setoptions(axes=boxed):
N:=64; f:=0.15; Freq:=500; N_band:=N/2;
Freq_max:=Freq/2; Step_band:=evalf(Freq_max/N_band);
A:=array(1..2);
Sig:=Vector(N,n->sin(n*2*Pi*f)):
A[1]:=listplot(Sig,color=magenta,title="Original Signal"):
freq:=FourierTransform(Sig, N);
Points_DFT:=[seq([i,evalc(Re(freq[i]))],i=1..N)]:
```

```

pointplot(Points_DFT,color=blue,thickness=3,title="DFT");
Points_Sp:=[seq([i*Step_band,abs(freq[i])],i=1..N_band)]:
plot(Points_Sp,title="Frequency Spectrum",color=green);
Sig_inv:=InverseFourierTransform(freq,N);
Points_inv:=[seq([i,Re(Sig_inv[i])],i=1..N)]:
A[2]:=plot(Points_inv,color=blue,title="Recovered Signal"):
display(A,tickmarks=[2,8]);

```

*Mathematica:*

```

{n=64, f=0.15, freq=500, nBand=n/2, freqMax=freq/2,
 stepBand=freqMax/nBand//N}
sig=Table[Sin[i*2*Pi*f],{i,1,n}]
g1=ListPlot[sig,PlotStyle->Magenta,PlotJoined->True,
             PlotLabel->"Original Signal",
             DisplayFunction->Identity];
f1=Fourier[sig]//Chop
pointsFFT=Table[{i,ComplexExpand[Re[f1[[i]]]]},{i,1,n}]
ListPlot[pointsFFT,PlotStyle->{PointSize[0.018],Blue},
          PlotLabel->"FFT",AspectRatio->1];
pointsSp=Table[{i*stepBand,Abs[f1[[i]]]}, {i,1,nBand}]
ListPlot[pointsSp,PlotStyle->Green,PlotJoined->True,
          PlotLabel->"Frequency Spectrum",PlotRange->All];
sigInv=InverseFourier[f1]
pointsInv=Table[{i,ComplexExpand[Re[sigInv[[i]]]]},{i,1,n}]
g2=ListPlot[pointsInv,PlotStyle->Blue,PlotJoined->True,
            PlotLabel->"Recovered Signal",
            DisplayFunction->Identity];
Show[GraphicsArray[{{g1},{g2}}],
     DisplayFunction->$DisplayFunction];

```

**Problem:** Generate a signal, obtain the fast Fourier transform, and recover the original signal.

*Maple:*

```

with(plots): with(LinearAlgebra):
setoptions(axes=boxed):
f:=0.15: n:=6; N:=2^n; A:=array(1..2);
Sig_Re:=Vector(N,[sin(2*Pi*f*i)$i=1..N]):
Sig_Im:=Vector(N,0):
A[1]:=listplot(Sig_Re,color=blue,title="Original Signal"):

```

```

FFT(n, Sig_Re, Sig_Im);
G1:=listplot(Sig_Re,color=magenta,title="FFT"):
iFFT(n, Sig_Re, Sig_Im);
A[2]:=listplot(Sig_Re,color=red,title="Recovered Signal"):
display(A,tickmarks=[2,8]); display(G1);

```

*Mathematica:*

```

{f=0.15, n=6, n1=2^n,
 sigRe=Table[Sin[2*Pi*f*i],{i,1,n1}]}
{f1=Fourier[sigRe]//Chop,
 f2=InverseFourier[f1]}
ListPlot[ComplexExpand[Re[f1]],PlotStyle->Magenta,
 PlotJoined->True,PlotLabel->"FFT"];
g1=ListPlot[sigRe,PlotStyle->Blue,PlotJoined->True,
 PlotLabel->"Original Signal",DisplayFunction->Identity];
g2=ListPlot[f2,PlotStyle->Red,PlotJoined->True,
 PlotLabel->"Recovered Signal",DisplayFunction->Identity];
Show[GraphicsArray[{{g1},{g2}}],
 DisplayFunction->$DisplayFunction];

```

**Problem:** Let  $f(x) = \exp(-(t - \pi))$  for  $t \in [0, 2\pi]$ . Find the Fourier polynomial of degree  $n$ .

*Maple:*

```

with(DiscreteTransforms): with(plots):
Digits:=15: N:=9: M:=2^N: S:=NULL: L:=Pi: K:=20;
DataRe:=Vector(M,0);
cA:=Vector(K+1,0); cB:=Vector(K+1,0);
f:=t->exp(-(t-Pi));
for i from 0 to M-1 do
    x:=evalf(2*Pi*i/M);
    S:=S,[x,evalf(f(x))];
od:
for i from 1 to M do DataRe[i]:=S[i][2]; od:
DataIm:=Vector(M,0): op(DataRe);
Sol:=evalf(2/sqrt(M)*FourierTransform(DataRe)):
op(Sol);
for i from 1 to K+1 do
    cA[i]:=Re(Sol[i]); cB[i]:=-Im(Sol[i]);
od:

```

```

F:=unapply(cA[1]/2+add(cA[i+1]*cos(X*i*Pi/L)
                         +cB[i+1]*sin(X*i*Pi/L), i=1..K), X);
G1:=plot(f(t), t=0..2*L, color=blue):
G2:=plot(F(X), X=0..2*L, view=[0..2*L, 0..23],
          color=magenta, thickness=3):
G:=array(1..2); G[1]:=G1; G[2]:=G2:
display(G, axes=boxed, scaling=constrained);
display([G1, G2], axes=boxed);

```

*Mathematica:*

```

f[t_]:=Exp[-(t-Pi)]; {n=9,m=2^n,s={},L=Pi}
For[i=0,i<=m-1,i++,
  x=N[2*Pi*i/m];
  s=Append[s,{x,N[f[x]]}]
];
data=Table[s[[i,2]],{i,1,m}]
sol=2/Sqrt[m]*Chop[Fourier[data]]
{k=20, cA=Re[Take[sol,k+1]], cB=Im[Take[sol,k+1]]}
F[x_]:=cA[[1]]/2+Sum[cA[[i+1]]*Cos[x*i*Pi/L]
                      +cB[[i+1]]*Sin[x*i*Pi/L],{i,1,k}];
g1=Plot[f[t],{t,0,2*L},PlotStyle->Hue[0.7],
         DisplayFunction->Identity];
g2=Plot[F[x],{x,0,2*L},PlotRange->{{0,2*L},{0,23}},
         PlotStyle->Hue[0.9],DisplayFunction->Identity];
Show[GraphicsArray[{{g1},{g2}}]];
Show[{g1,g2},DisplayFunction->$DisplayFunction,
      PlotRange->{{0,2*L},{0,23}}];

```

In *Mathematica*, with the package `Calculus`FourierTransform``, the numerical approximations to the Fourier transforms can be calculated. This package also includes functions for the Fourier series, Fourier coefficients, discrete-time Fourier transforms (in more detail, see `?Calculus`FourierTransform`*`).

In this package the options of the function `NIntegrate` and the option `FourierParameters` are valid. Here we discuss the most important functions:

```
<<Calculus`FourierTransform`;
NFourierTransform[expr,x,t]
DTFourierTransform[expr,x,t]
NDTFourierTransform[expr,x,t]
NInverseFourierTransform[expr,t,x]
InverseDTFourierTransform[expr,t,x]
NInverseDTFourierTransform[expr,x,t]
```

```
<<Calculus`FourierTransform`;
expr=Exp[-t^4]
FourierTransform[expr,t,s]//Timing
NFourierTransform[expr,t,-10]//Timing
points=Table[NFourierTransform[expr,t,i],{i,-20,20}]
ListPlot[Abs[points],PlotJoined->True,
PlotStyle->{Thickness[0.015],Blue},
PlotRange->All,Frame->True];
```

## 9.4 Hankel Transforms

The *Hankel transforms* of order zero ( $n = 0$ ) and of order one ( $n = 1$ ) are useful for solving initial-value and boundary-value problems involving the Laplace or Helmholtz equations in an axisymmetric cylindrical geometry.

In *Maple*, the Hankel transforms can be obtained with the function

`hankel` according to the formula  $F(s) = \int_0^\infty f(t)J_\nu(st)\sqrt{st} dt$  and the package `inttrans`.

In *Mathematica*, there is no built-in function for constructing Hankel transforms. We define the function `HankelTransform` according to the formula  $F(s) = \int_0^\infty f(t)J_\nu(st)t dt$ .

**Problem:** Find the Hankel transforms for the functions  $f(r), \frac{\delta(a-r)}{r}$ .

*Maple:*

```
with(inttrans); assume(a>0);
convert(hankel(f(r),r,k,0),int); hankel(Dirac(a-r)/r,r,k,0);
```

*Mathematica:*

```
HankelTransform[f_,t_,s_,nu_,assump_List]:=Module[{},
  FullSimplify[Integrate[f[t]*BesselJ[nu,s*t]*t,{t,0,Infinity},
    Assumptions->{t>0}],assump]
f2[r_]:=DiracDelta[a-r]/r;
{HankelTransform[f1,r,k,0,{a>0,k>0}],
 HankelTransform[f2,r,k,0,{a>0,k>0}]}
```

The *Hankel transforms* of unknown functions (for *Maple*) can be added with the function `addtable`. For example, to add the Hankel transform for  $\exp(-ar)$ .

*Maple:*

```
with(inttrans);
assume(a>0); hankel(exp(-a*r),r,k,0);
addtable(hankel,exp(-a*r),k*(a^2+k^2)^(-3/2),r,k,
          hankel=nu::Range(-infinity,infinity));
hankel(exp(-a*r),r,k,0);
```

**Problem:** The Hankel transform is self-inverting for  $n > -1/2$ . Find the forward and inverse Hankel transforms for  $1/\sqrt{r}$ .

*Maple:*

```
with(inttrans);
F:=hankel(r^(-1/2),r,k,0); hankel(F,k,r,0);
```

*Mathematica:*

```
f[r_]:=1/Sqrt[r];
HankelTransform[f_,t_,s_,nu_,assump_List:{}]:=Module[{},
  FullSimplify[Integrate[f[t]*BesselJ[nu,s*t]*t,{t,0,Infinity}],
  Assumptions->{t>0},assump]];
HankelTransformOrden0[f_,t_,s_,assump_List:{}]:=Module[{},
  FullSimplify[Integrate[
    FullSimplify[2*Pi*f[t]*BesselJ[0,2*Pi*s*t]*t],{t,0,Infinity},
    Assumptions->{t>0},assump]];
HankelTransform[f,r,k,0,{r\[Element] Reals,r>0,k>0}]
HankelTransform[f,k,r,0,{r\[Element] Reals,r>0,k>0}]
HankelTransformOrden0[f,r,k,{r\[Element] Reals,r>0,k>0}]
HankelTransformOrden0[f,k,r,{r\[Element] Reals,r>0,k>0}]
```

*Applications* to partial differential equations.

**Problem:** Obtain the solution of the free vibration of a large circular membrane described by the initial-value problem:

$$\begin{aligned} u_{tt} &= c^2(u_{rr} + u_r/r), \quad 0 \leq r < \infty, \quad t > 0, \\ u(r, 0) &= f(r), \quad u_t(r, 0) = g(r), \quad 0 \leq r \leq \infty, \end{aligned}$$

where  $f(r)$  and  $g(r)$  are arbitrary functions (use the *Maple* function `hankel`).

*Maple:*

```
with(inttrans): with(PDEtools):
declare(u(r,t),U(t)); ON;
Eq1:=c^2*(diff(u(r,t),r$2)+1/r*diff(u(r,t),r))
      =diff(u(r,t),t$2);
Eq2:=hankel(Eq1,r,k,0);
Eq3:=subs({hankel(u(r,t),r,k,0)[t,t]=diff(U(t),t$2),
            hankel(u(r,t),r,k,0)=U(t)},Eq2);
IC1:={u(r,0)=f(r),D[2](u)(r,0)=g(r)};
IC2:=hankel(IC1,r,k,0);
IC3:=subs({hankel(D[2](u)(r,0),r,k,0)=D(U)(0),
            hankel(u(r,0),r,k,0)=U(0),hankel(f(r),r,k,0)=F(k),
            hankel(g(r),r,k,0)=G(k)},IC2);
sys:={Eq3} union IC3;
Sol := dsolve(sys, U(t));
Sol_Fin:=expand(convert(hankel(rhs(Sol),k,r,0),int));
```

# Chapter 10

## Mathematical Equations

### 10.1 Algebraic and Trascendental Equations

*Exact solutions* of algebraic equations or systems of equations.

*Maple*:

`solve`, `RootOf`, representing for roots of equations;  
`allvalues`, computing all possible values of expressions that contain  
`RootOfs`;  
`isolate`, isolating a subexpression to left side of an equation.

```
solve(Eq, var);    solve({Eq1, Eq2}, {var1, var2});  
RootOf(expr,x);allvalues(expr,ops);isolate(Eq,expr);
```

*Mathematica*:

`Solve`, `Root`, `Roots`, representing for roots of equations;  
`Reduce`, all possible solutions or verifying identities;  
`Algebra`RootIsolation``, counting and isolating roots of polynomials.

```
Solve[eq, var]      Solve[{eq1, eq2, ...}, {var1, var2, ...}]  
Solve[eq, var, VerifySolutions->False] Reduce[eqs, vars]  
Root[eq, var]       Roots[eq, var]  
<<Algebra`RootIsolation`      CountRoots[f, {x, x1, x2}]  
RealRootIntervals[f]        ComplexRootIntervals[f]
```

We note that the roots obtained by `Solve` are expressed as a list of transformation rules  $x \rightarrow x_1$ . This notation indicates that the solution  $x=x_1$ , but  $x$  is not replaced by the value  $x_1$ . For accessing the values of the solutions, we can use the replacement operator ( $/.$ ) or list operation functions `Part` or `[[]]`.

**Problem:** Find the exact solutions of the equations:  $3x + 11 = 5$ ,  $x^3 - 5x^2 + 2x - 1 = 0$ ,  $\cos^2 x - \cos x - 1 = 0$ ,  $(x^2 - 1)/(x^2 + 5) + 1 = 0$ . Find the inverse function of  $f(x) = (2x - 3)/(1 - 5x)$ . Solve the equation  $V = \pi r^2/h$  for  $h$ .

*Maple:*

```
solve(3*x+11=5,x);           solve((x^2-1)/(x^2+5)+1=0,x);
solve(x^3-5*x^2+2*x-1=0,x); solve(cos(x)^2-cos(x)-1=0,x);
F_inv:=unapply(solve(y=(2*x-3)/(1-5*x), x), y); F_inv(x);
solve(v=Pi*r^2/h, h);       isolate(v=Pi*r^2/h, h);
```

*Mathematica:*

```
{Solve[3*x+11==5], Solve[Cos[x]^2-Cos[x]-1==0]}
sol1=Solve[x^3-5*x^2+2*x-1==0] // Simplify
Product[sol1[[i,1,2]],{i,1,3}] // Simplify
fInv[t_]:=Solve[y==(2*x-3)/(1-5*x),x]/.y->t; fInv[x]
{Solve[v==Pi*r^2/h, h], Reduce[v==Pi*r^2/h, h]}
Solve[Sqrt[x]+x==1,x,VerifySolutions->False]
Reduce[x^2-1==-(x-1)*(x+1),x]
Reduce[x^5-x^3+2*x-10==0 && x\[Element]Reals, x]/N
```

**Problem:** Solve the systems of equations:

$$\begin{cases} x^2+y^2=4, \\ y=2x, \end{cases} \quad \begin{cases} 2x - 3y + 4z=2, \\ 3x - 2y + z=0, \\ x + y + z=1. \end{cases}$$

Find the values of the expression  $\sin^2(x) + \cos^2(y)$  at the solution points.

*Maple:*

```
Sys1:={x^2+y^2=4,y=2*x}; Var1:={x,y};
Sol1:=evalf(allvalues(solve(Sys1,Var1)));
Sys2:={2*x-3*y+4*z=2,3*x-2*y+z=0,x+y+z=1};
Var2:={x,y,z}; Sol2:=solve(Sys2, Var2);
```

```
Expr:=sin(x)^2+cos(y)^2;
[[evalf(subs(Sol1[1],Expr)),
evalf(subs(Sol1[2],Expr))],subs(Sol2,Expr)];
```

*Mathematica:*

```
sys1={x^2+y^2==4,y==2*x}; var1={x,y};
sol1=N[Solve[sys1,var1],10]
sys2={2*x-3*y+4*z==2,3*x-2*y+z==0,x+y+z==1};var2={x,y,z};
sol2=Solve[sys2,var2]
Sin[x]^2+Cos[y]^2 /. {sol1,sol2}
```

*Numerical solutions* of algebraic and trascendental equations.

*Maple:*

```
evalf(solve(Eq,var));      fsolve(Eq,var,ops);
fsolve(Eq,var=a..b,ops); fsolve(Eq,var,complex);
```

*Mathematica:*

```
NSolve[eqs,vars]          NSolve[eqs,vars,n]
N[Solve[eqs,vars],n]
FindRoot[eq,{x,x0}]       FindRoot[eq,{x,{x0,x1}}]
FindRoot[eq,{x,x0,a,b}]   FindRoot[eq,{x,{x0,x1},a,b}]
FindRoot[eq,{x,I},MaxIterations->n,
         Jacobian->derivative, WorkingPrecision->n,
         DampingFactor->n]
FindRoot[eqs,{x1,x10},{x2,x20},...]
<<NumericalMath`InterpolateRoot`;
InterpolateRoot[eq,{x,x0,x1}]
```

Here  $x_0, x_1$  are the initial values of the variable  $x$  and  $[a, b]$  is the interval outside of which the iteration process stops.

The function `FindRoot` solves equations using iterative methods (Newton's method, the secant method) and has many options (see `Options[FindRoot]`).

**Problem:** Approximate the values of  $x$  that satisfy the equations  $x^5 - 2x^2 = 1 - x$ ,  $1 - x^2 = x^3$ .

*Maple:*

```
map(evalf,[solve(x^5-2*x^2=1-x, x)]);
map(evalf,[solve(1-x^2=x^3, x)]);
```

*Mathematica:*

```
{NSolve[x^5-2*x^2==1-x,x,20], N[Solve[x^5-2*x^2==1-x,x],20],
 NSolve[1-x^2==x^3,x], N[Solve[1-x^2==x^3,x]]}
```

**Problem:** Find the numerical solutions of the equation  $5x^5 - 4x^4 - 3x^3 + 2x^2 - x - 1 = 0$ . Approximate the roots of  $\cos(\sin x) - x^2 = 0$ . Find a numerical approximation to the solution of the equation  $\sin x = x/2$  for  $x \in (0, \pi]$ .

*Maple:*

```
Eq1:=5*x^5-4*x^4-3*x^3+2*x^2-x-1=0;
fsolve(Eq1, x); fsolve(Eq1, x, complex);
f:=x->cos(sin(x))-x^2; plot(f(x), x=-1..1);
S1:=fsolve(f(x), x=-1..0); S2:=fsolve(f(x), x=0..1);
fsolve(sin(x)=x/2, x=0..Pi, avoid={x=0});
```

*Mathematica:*

```
eq1=5*x^5-4*x^4-3*x^3+2*x^2-x-1==0
{FindRoot[eq1, {x,0}, MaxIterations->30],
 FindRoot[eq1, {x,I}], FindRoot[eq1, {x,-I}],
 FindRoot[eq1, {x,1}], FindRoot[eq1, {x,-0.5}]}
f[x_]:=Cos[Sin[x]]-x^2; Plot[f[x],{x,-1,1}];
sol1=FindRoot[f[x],{x,-0.9,-1,0},WorkingPrecision->30]
sol2=FindRoot[f[x],{x,0.1,0,1},DampingFactor->2]
FindRoot[Sin[x]==x/2,{x,0.0001,Pi}]
```

**Problem:** Find exactly and approximately the intersection points of the graphs  $f_1(x) = x^2 + x + 10$  and  $f_2(x) = 9x^2$ .

*Maple:*

```
f1:=x->x^2+x+10; f2:=x->9*x^2;
plot({f1(x),f2(x)},x=-3..3); sols:=[solve(f1(x)=f2(x),x)];
for i from 1 to nops(sols) do
  evalf(sols[i]); evalf(f1(sols[i]));
  evalf(f2(sols[i]));
od;
```

*Mathematica:*

```
f1[x_]:=x^2+x+10; f2[x_]:=9*x^2;
Plot[{f1[x],f2[x]},{x,-3,3}];
sols=Solve[f1[x]==f2[x],x]; {x,f1[x]}/.sols//Simplify//N
```

**Problem:** Find the numerical solutions of  $\begin{cases} \sin x - \cos y = 1, \\ e^x - e^{-y} = 1. \end{cases}$

*Maple:*

```
with(plots):
sys:={sin(x)-cos(y)=1,exp(x)-exp(-y)=1};
implicitplot(sys,x=-Pi..Pi,y=-Pi..Pi);
fsolve(sys,{x=0,y=3}); fsolve(sys,{x=0,y=-3});
fsolve(sys,{x=2,y=-3});
```

*Mathematica:*

```
<<Graphics`ImplicitPlot`;
ImplicitPlot[{Sin[x]-Cos[y]==1,Exp[x]-Exp[-y]==1},
{x,-Pi,Pi},{y,-Pi,Pi},AspectRatio->1];
FindRoot[{Sin[x]-Cos[y]==1,Exp[x]-Exp[-y]==1},{x,2},{y,-2}]
```

*Approximate analytical solutions* of algebraic equations. In *Maple* and *Mathematica* there is no single function for finding approximate analytical solutions to equations, so various methods can be applied or developed.

**Problem:** Find approximate analytical solutions to the algebraic equation  $F(x; \varepsilon) = 0$ , where  $F$  is a real function of  $x$  and a small parameter  $\varepsilon$ .

According to the regular perturbation theory, the equation  $F(x; 0) = 0$  has a solution  $x_0$  and the solution of the perturbed equation is near  $x_0$  and can be represented as a power series of  $\varepsilon$ .

For example, we solve the equation  $x^2 - ax + \varepsilon = 0$ ,  $|\varepsilon| \ll 1$ . If  $\varepsilon = 0$ , the roots are  $x_0 = 0, a$ . If  $\varepsilon \rightarrow 0$ , we have  $X_1(\varepsilon) \rightarrow 0$ ,  $X_2(\varepsilon) \rightarrow a$ . If  $|\varepsilon| \ll 1$ , the roots are:  $X_i = x_0 + \varepsilon x_1 + \dots + \varepsilon^k x_k + \dots$ ,  $i = 1, 2, k = 1, 2, \dots$ , where  $x_k$  are the unknown coefficients to be determined. Substituting the series  $X_i$  into the original equation and matching the coefficients of like powers of  $\varepsilon$ , we arrive to a system of algebraic equations for the  $i$ -th approximation, which can be solved for  $x_k$ .

*Maple:*

```
RegPertPoly := proc(Expr, var, param)
  global Sers: local i, j, Expr1, X, y, k, m:
  y := var[0]; Expr1:=collect(Expr(y),param);
  X[0]:=[solve(coeff(Expr1,param,0),var[0])]; k:=nops(X[0]);
  for j from 1 to k do y[j]:=X[0][j];
    for i from 1 to n do
      y[j]:=y[j]+param^i*var[i]; Expr1:=Expr(y[j]);
      X[i]:=solve(coeff(Expr1,param,i),var[i]);
      y[j]:=X[0][j]+add(X[m]*param^m,m=1..i);
    od:
  Sers:=sort(convert(y,list)):
  od: RETURN(Sers): end:
n:=5; Eq:=x->x^2-a*x+epsilon; RegPertPoly(Eq,x,epsilon);
```

*Mathematica:*

```
regPertPoly[f_,var_,param_]:=Module[{y,expr,X0,k,m,i,j,X},
  y=var[0]; expr=Collect[f[var[0]],param];
  X0=Solve[Coefficient[expr,param,0]==0,y]; k=Length[X0];
  For[j=1,j<=k,j++, y[j]=X0[[j,1,2]];
    For[i=1,i<=n,i++, y[j]=y[j]+param^i*var[i]; expr=f[y[j]];
      X[i]=Solve[Coefficient[expr,param,i]==0,var[i]];
      y[j]=X0[[j,1,2]]+Sum[X[m][[1,1,2]]*param^m,{m,1,i}];
    ];
    serSol=Table[y[j],{j,1,k}];
  ]; Return[serSol];
n=5; eq[x_]:=x^2-a*x+\[Epsilon]; regPertPoly[eq,x,\[Epsilon]]
```

## 10.2 Ordinary Differential Equations

In *Maple*, there exists a large set of functions to solve (analytically, numerically, graphically) ordinary and partial differential equations or the systems of differential equations. We discuss the most important functions and packages.

*Exact solutions* to ordinary differential equations.

*Maple*:

- `dsolve`, find closed form solutions for a single ODE or a system of ODEs (see `?dsolve`);
- `dsolve,{ODEs,ICs}`, solve ODEs or a system of them with given initial conditions;
- `dsolve,formal_series`, find formal power series solutions to a homogeneous linear ODE with polynomial coefficients;
- `dsolve,series`, find series solutions to ODEs problems;
- `dsolve,method`, find solutions using integral transforms;
- `dsolve[interactive]`, interactive symbolic and numeric solving of ODEs, etc.

```
dsolve(ODE,y(x),ops); dsolve({ODEs},{funcs});
dsolve({ODEs},ICs,{funcs},ops);
dsolve(ODE,y(x),'formal_series','coeffs'=coeff_type);
dsolve({ODEs},ICs,{funcs},'series');
dsolve({ODEs},ICs,{funcs}, method=transform, ops);
dsolve[interactive]({ODEs},ops);
```

*Mathematica*:

- `DSolve`, find the general solution  $y[x]$  or  $y$  (expressed as a “pure” function) for a single ODE or a system of ODEs;
- `DSolve,{ODEs,ICs}`, solve ODEs or a system of them with given initial or boundary conditions.

<code>DSolve[ODE,y[x],x]</code>	<code>DSolve[ODE,y,x]</code>
<code>DSolve[{ODEs,ICs},y[x],x]</code>	<code>DSolve[{ODEs,ICs},y,x]</code>

*Explicit, implicit forms* of the exact solutions, graphs of solutions. General solutions.

*Maple:*

```
with(plots):
Sol_Ex := dsolve(diff(y(t),t)+t^2/y(t)=0, y(t));
Sol_Imp:= dsolve(diff(y(t),t)+t^2/y(t)=0, y(t),implicit);
G:=subs({y(t)=y},lhs(Sol_Imp));
Gs:=seq(subs(_C1=i,G),i=-5..5);
contourplot({Gs},t=-5..5,y=-10..10,color=blue);
```

*Mathematica:*

```
<<Graphics`ImplicitPlot`;
DSolve[y'[t]+t^2/y[t]==0,y[t],t]
sol=DSolve[y'[t]+t^2/y[t]==0,y,t]
y'''[t]+y[t]/.sol
f:=sol[[1,1,2]]; {f[x],f'[x],f''[x]}
G=Table[y^2==(f[t])^2/.{C[1]->i},{i,-5,5}]
ImplicitPlot[G,{t,-5,5},
    PlotRange->{-5,5},PlotStyle->Hue[0.7]];
DSolve[D[y[t],{t,2}]+y[t]==0,y[t],t,GeneratedParameters->A]
```

*ODE classification* and solution methods suggestion, e.g., separation of variables, homogeneous equations, series solutions, exact equations, linear equations, etc., see `?DEtools`, `?odeadvisor`.

*Maple:*

```
with(DEtools):
ODE1:=diff(y(t),t)=y(t)*sin(t)^2/(1-y(t));
ODE2:=(t^2-y(t)*t)*diff(y(t),t)+y(t)^2=0;
S_expl:=dsolve(ODE1,y(t));
S_impl:=dsolve(ODE1,y(t),implicit);
odeadvisor(ODE1);
Sol_sep:=separablesol(ODE1,y(t));
odeadvisor(ODE2);
Sol1:=dsolve(ODE2,y(t));
Sol2:=genhomosol(ODE2,y(t));
Sol_ser:=dsolve(ODE1,y(t),'series');
```

*Higher-order ODE:* exact and numeric solutions, and their graphs.

*Maple:*

```
with(plots):
setoptions(axes=boxed,scaling=constrained,numpoints=200);
ODE:=diff(x(t),t$2)-diff(x(t),t)+(t-1)*x(t)=0;
ICs:=D(x)(0)=0,x(0)=1;
Sol_ex:=dsolve({ODE,ICs},x(t));
Sol_num:=dsolve({ODE,ICs},x(t),numeric);
G:=array(1..3);
G[1]:=odeplot(Sol_num,[t,x(t)],0..10,color=blue):
G[2]:=plot(rhs(Sol_ex),t=0..10,color=red):
G[3]:=odeplot(Sol_num,[x(t),diff(x(t),t)],0..10,
               color=magenta):
display(G);
```

*Mathematica:*

```
ODE={x''[t]-x'[t]+(t-1)*x[t]==0};
ICs={x'[0]==0,x[0]==1};
eq1=DSolve[{ODE,ICs},x[t],t]
g1=Plot[x[t]/.eq1,{t,0,10},PlotStyle->{Hue[0.5],
Thickness[0.01]},DisplayFunction->Identity];
eq2=DSolve[{ODE,ICs},x,t]; solEx=eq2[[1,1,2]];
Plot[solEx[t],{t,0,10},PlotStyle->{Hue[0.6],
Thickness[0.01]}];
eq3=NDSolve[{ODE,ICs},x,{t,0,10}];
solN=eq3[[1,1,2]]
Table[{t,solN[t]},{t,0,10}]/.TableForm
g2=Plot[solN[t],{t,0,10},PlotStyle->{Hue[0.8],
Thickness[0.01]},DisplayFunction->Identity];
g3=Plot[solN'[t],{t,0,10},PlotStyle->{Hue[0.9],
Thickness[0.01]},DisplayFunction->Identity];
Show[g1,g3,DisplayFunction->$DisplayFunction];
Show[g2,g3,DisplayFunction->$DisplayFunction];
ParametricPlot[{solN[t],solN'[t]},{t,0,10},
PlotStyle->{Hue[0.7]}];
```

*ODE systems:* exact solutions, the integral Laplace transforms (in more detail, see Sect. 9.1), graphs of solutions.

*Maple:*

```
with(DEtools): with(inttrans):
ODE_sys1:={D(x)(t)=-2*x(t)+5*y(t),D(y)(t)=4*x(t)-3*y(t)};
Sol_sys1:=dsolve(ODE_sys1, {x(t),y(t)});
A1 := array([[-2,5],[4,-3]]); matrixDE(A1,t);
ODE_sys2:={diff(x(t),t)=-y(t)+cos(2*t),
           diff(y(t),t)=5*x(t)+2*sin(2*t)};
Eq1:=laplace(ODE_sys2,t,p); Eq2:=subs({x(0)=2,y(0)=0},Eq1);
Eq3:=solve(Eq2,{laplace(x(t),t,p),laplace(y(t),t,p)});
Sol_sys2:=invlaplace(Eq3,p,t); assign(Sol_sys2):
plot([x(t),y(t),t=-3..3],color=blue);
```

*Mathematica:*

```
<<Graphics`PlotField`;
sys1={x'[t]==-2*x[t]+5*y[t],y'[t]==4*x[t]-3*y[t],
      x[0]==1,y[0]==0}
solsys1=DSolve[sys1,{x[t],y[t]},t]
sys2={x'[t]==-y[t]+Cos[2*t],y'[t]==5*x[t]+2*Sin[2*t]}
eq1=LaplaceTransform[sys2,t,p]/.{x[0]->2,y[0]->0}
eq2=Solve[eq1,{LaplaceTransform[x[t],t,p],
              LaplaceTransform[y[t],t,p]}]
xsol=InverseLaplaceTransform[eq2[[1,1,2]],p,t]
ysol=InverseLaplaceTransform[eq2[[1,2,2]],p,t]
ParametricPlot[{xsol,ysol},{t,-3,3},PlotStyle->Hue[0.5]];
```

*Numerical and graphical solutions* to ordinary differential equations.

*Maple:*

- dsolve[numeric]**, find numerical solutions to ODEs problems;
- odeplot**, graphs or animations of 2D and 3D solution curves obtained from the numerical solution;
- phaseportrait**, phase portraits for a system of first order differential equations or a single higher order differential equation with initial conditions;
- DEplot**, vector fields for autonomous systems of first order differential equations, etc.

```
with(plots); with(DEtools);
dsolve({ODEs}, numeric, {funcs}, ops);
dsolve(numeric, {funcs}, procops, ops);
dsolve({ODEs}, numeric, output=operator);
NS := dsolve({ODEs},numeric,{funcs},ops);
odeplot(NS, {funcs}, range, ops);
phaseportrait({ODEs}, {funcs}, range, {ICs}, ops);
DEplot({ODEs}, {funcs}, trange, ops);
```

*Mathematica:*

**NDSolve**, find numerical solutions to ODEs problems;  
**PlotVectorField**, construct vector fields for autonomous systems of first order differential equations (see Sect. 5.10);  
**LaplaceTransform**, find solutions using integral transforms (in more detail, see Sect. 9.1).

```
NDSolve[{ODEs,ICs}, y[t], {t,t1,t2}]
NDSolve[{ODEs,ICs}, y, {t,t1,t2}]
```

*Initial value problems*, graphs of solutions.

*Maple:*

```
with(plots):
N:=7; Sols:=Vector(N,0): Gr:=NULL:
IVP:={D(y)(t)=3*t+2*y(t),y(0)=n}; Sol:=dsolve(IVP,y(t));
for i from 1 to N do
  Sols[i]:=subs(n=-3+(i-1),Sol);
od: op(Sols);
SList:=[`rhs(Sols[i])` $ 'i'=1..N];
for i from 1 to N do
  G||i:=plot(SList[i], t=0..2.5, color=blue):
  Gr:= Gr,G||i:
od:
VField:=fieldplot([1,3*t+2*y],t=0..2.5,y=-600..600,
                  grid=[30,30],arrows=slim, color=t):
display({Gr,VField},axes=boxed);
```

*Mathematica:*

```
<<Graphics`PlotField`;
IVP={y'[t]==3*t+2*y[t],y[0]==n}
sols=Table[DSolve[IVP,y[t],t],{n,-3,3}]
slist=Table[sols[[i,1,1,2]],{i,1,7}]
Do[g[i]=Plot[slist[[i]],{t,0,2.5},PlotRange->All,
              DisplayFunction->Identity,
              PlotStyle->Hue[0.7]],{i,1,7}];
gr=Table[g[i],{i,1,7}];
vField=PlotVectorField[{1,3*t+2*y},{t,0,2.5},{y,-500,500},
                       Axes->Automatic,PlotStyle->Hue[0.8],
                       HeadLength->0,AspectRatio->1,
                       DisplayFunction->Identity];
Show[gr,vField,DisplayFunction->$DisplayFunction];
```

*Boundary value problems,* graphs of solutions.

*Maple:*

```
BVP:={diff(y(t),t$2)+2*y(t)=0,y(0)=1,y(Pi)=0};
Sol:=dsolve(BVP,y(t));
plot(rhs(Sol),t=0..Pi,axes=boxed,color=blue,
      thickness=3);
```

*Mathematica:*

```
BVP={y''[t]+2*y[t]==0,y[0]==1,y[Pi]==0}
sol=DSolve[BVP,y[t],t]
Plot[sol[[1,1,2]],[t,0,Pi],Frame->True,Axes->False,
      PlotStyle->{Hue[0.7],Thickness[0.01]}];
```

**Problem:** Solve the linear system of ordinary differential equations with constant coefficients

$$\begin{cases} x' = x + 3y, \\ y' = -2x + y, \end{cases}$$

and graph the solutions together with the direction field associated with the system.

*Maple:*

```
with(linalg): with(DEtools): with(plots): with(student):
A:=array([[1,3],[-2,1]]): eigenvectors(A); matrixDE(A,t);
ODE_sys:=equate(array([[diff(x(t),t)],[diff(y(t),t)]]),
A &* array([[x(t)],[y(t)]]));
Sol_sys:=dsolve(ODE_sys, {x(t),y(t)}); assign(Sol_sys);
Curves:={seq(seq(subs({_C1=i,_C2=j},[x(t),y(t),t=-3..3]),
i=-3..3),j=-3..3)}:
G1:=plot(Curves,view=[-10..10,-10..10],color=blue,
axes=boxed): unassign('x', 'y');
G2:=DEplot(ODE_sys, [x(t),y(t)],t=-3..3,x=-10..10,
y=-10..10,color=red): display({G1, G2});
```

*Mathematica:*

```
<<Graphics`PlotField`;
sys3={x'[t]==x[t]+3*y[t],y'[t]==-2*x[t]+y[t]};
sol=Table[DSolve[sys3,{x[t],y[t]},t]/.{C[1]->i,C[2]->j},
{i,1,5},{j,1,5}];
g1=Table[ParametricPlot[{sol[[i,j,1,1,2]],sol[[i,j,1,2,2]]},
{t,-3,3},PlotStyle->Hue[0.9],Frame->True,
Background->Hue[0.6],DisplayFunction->Identity],
{i,1,5},{j,1,5}];
g2=PlotVectorField[{x+3*y,-2*x+y},{x,-30,30},{y,-30,30},
PlotStyle->Hue[0.7],HeadLength->0,
DisplayFunction->Identity];
Show[g1,g2,DisplayFunction->$DisplayFunction];
```

**Problem:** Solve numerically the initial value problem

$$x' = xy, \quad y' = x + y, \quad x(0) = 1, \quad y(0) = 1,$$

and graph the solutions.

*Maple:*

```
with(plots): setoptions(scaling=constrained);
A:= array(1..3); IC:={x(0)=1,y(0)=1};
ODE:={D(x)(t)=x(t)*y(t),D(y)(t)=x(t)+y(t)};
Sol:=dsolve(ODE union IC,numeric,output=operator);
```

```
A[1]:=plot(rhs(Sol[2](t)),t=0..1):
A[2]:=plot(rhs(Sol[3](t)),t=0..1):
A[3]:=plot({rhs(Sol[2](t)),rhs(Sol[3](t))},t=0..1):
display(A);
```

*Mathematica:*

```
ODE={x'[t]==x[t]*y[t],y'[t]==x[t]+y[t]}
ICs={x[0]==1,y[0]==1}
eq1=NDSolve[{ODE,ICs},{x,y},{t,0,1}];
{solNX=eq1[[1,1,2]], solNY=eq1[[1,2,2]]}
Table[{t,solNX[t]},{t,0,1,0.1}]// TableForm
Table[{t,solNY[t]},{t,0,1,0.1}]// TableForm
g1=Plot[solNX[t],{t,0,1},PlotStyle->{Hue[0.8],
Thickness[0.01]},DisplayFunction->Identity];
g2=Plot[solNY[t],{t,0,1},PlotStyle->{Hue[0.9],
Thickness[0.01]},DisplayFunction->Identity];
g12=Show[g1,g2,DisplayFunction->Identity];
Show[GraphicsArray[{{g1},{g2},{g12}}],Frame->True,
DisplayFunction->$DisplayFunction,AspectRatio->1];
```

**Problem:** Solve the nonlinear initial value problem

$$y' = -e^{yt} \cos(t^2), \quad y(0) = p.$$

Graph the solutions  $y(t)$  for various values of the parameter  $p$  on the interval  $[0, \pi]$ . Graph the solutions for various initial conditions  $p = 0.1i$  ( $i = 1, 2, \dots, 5$ ).

*Maple:*

```
with(plots): NGSol:=proc(IC)
local Eq,Eq_IC,L1,Sol_N,N_IC,i; global P;
Eq:=D(y)(t)=-exp(y(t)*t)*cos(t^2); L1:=NULL;
N_IC:=nops(IC);
for i from 1 to N_IC do
Eq_IC:=evalf(y(0)=IC[i]);
Sol_N:=dsolve({Eq,Eq_IC},y(t),type=numeric,range=0..P);
L1:=L1,odeplot(Sol_N,[t,y(t)],0..P,numpoints=100,
color=blue,thickness=2,axes=boxed): od;
display([L1]); end:
List1:=[seq(0.1*i,i=1..5)]; P:=evalf(Pi); NGSol(List1);
```

*Mathematica*:

```
eq=y'[t]==-Exp[y[t]*t]*Cos[t^2];
Do[{ICs={y[0]==0.1*i};
sN=NDSolve[{eq,ICs},y[t],{t,0,Pi},MaxSteps->1000];
solN[t_]:=sN[[1,1,2]];
g[i]=Plot[solN[t],{t,0,Pi},PlotStyle->Hue[0.5+i*0.07],
DisplayFunction->Identity];}, {i,1,10}]
Show[g[1],g[2],g[3],g[4],g[5],
DisplayFunction->$DisplayFunction];
```

**Problem:** Construct a phase portrait of the dynamical system that describes the evolution of the amplitude and the slow phase of a fluid under the subharmonic resonance

$$\frac{dv}{dt} = -\nu v + \varepsilon u \left[ \delta + \frac{1}{4} - \frac{1}{2} \phi_2(u^2 + v^2) + \frac{1}{4} \phi_4(u^2 + v^2)^2 \right],$$

$$\frac{du}{dt} = -\nu u + \varepsilon v \left[ -\delta + \frac{1}{4} + \frac{1}{2} \phi_2(u^2 + v^2) - \frac{1}{4} \phi_4(u^2 + v^2)^2 \right].$$

Note that this system has been obtained in [32] by averaging transformations with *Maple*. Here  $\nu$  is the fluid viscosity,  $\varepsilon$  is the small parameter,  $\phi_2, \phi_4$  are the second and the fourth corrections to the nonlinear wave frequency,  $\delta$  is the off resonance detuning. Choosing the corresponding parameter values (for the six regions where the solution exists), we can obtain a phase portrait.

*Maple*:

```
with(plots): with(DEtools): delta:=-1/2;
phi_2:=1; phi_4:=1; nu:=0.005; epsilon:=0.1;
Eq1 := D(v)(t)=-nu*v(t)+epsilon*u(t)*(delta + 1/4
-phi_2/2*(u(t)^2+v(t)^2)+phi_4/4*(u(t)^2+v(t)^2)^2);
Eq2 := D(u)(t)=-nu*u(t)+epsilon*v(t)*(-delta + 1/4
+phi_2/2*(u(t)^2+v(t)^2)-phi_4/4*(u(t)^2+v(t)^2)^2);
Eqs := [Eq1, Eq2]; vars := [v(t), u(t)];
IC := [[u(0)=0, v(0)=1.1033], [u(0)=0, v(0)=-1.1033],
[u(0)=1.1055, v(0)=0], [u(0)=-1.1055, v(0)=0],
[u(0)=0, v(0)=1.613], [u(0)=0, v(0)=-1.613]];
Ops := arrows=medium, dirgrid=[20, 20], stepsize=0.1,
thickness=2, linecolour=blue, color=green;
phaseportrait(Eqs, vars, t=-48..400, IC, Ops);
```

*Mathematica:*

```
<<Graphics`PlotField`;
delta=-1/2; phi2=1; phi4=1; nu=0.005; epsilon=0.1;
eq1=-nu*v[t]+epsilon*u[t]*(delta+1/4
    -phi2/2*(u[t]^2+v[t]^2)+phi4/4*(u[t]^2+v[t]^2)^2);
eq2=-nu*u[t]+epsilon*v[t]*(-delta+1/4
    +phi2/2*(u[t]^2+v[t]^2)-phi4/4*(u[t]^2+v[t]^2)^2);
ICs={{0,1.1033},{0,-1.1033},{1.1055,0},{-1.1055,0},
    {0,1.613},{0,-1.613}};
n=Length[ICs];
Do[{sys[i]={v'[t]==eq1,u'[t]==eq2,
    v[0]==ICs[[i,1]],u[0]==ICs[[i,2]]},
    sols=NDSolve[sys[i],{v,u},{t,-48,400}];
    cv=v/.sols[[1]]; cu=u/.sols[[1]];
    c[i]=ParametricPlot[Evaluate[{cv[t],cu[t]}],{t,-48,400},
        PlotStyle->{Hue[0.1*i+0.2],Thickness[.008]},
        DisplayFunction->Identity];}, {i,1,n}]
fv=eq1/.{v[t]->v,u[t]->u};
fu=eq2/.{v[t]->v,u[t]->u};
fd=PlotVectorField[{fv,fu},{v,-2.2,2.2},{u,-2.2,2.2},
    Frame->True,HeadLength->0,DisplayFunction->Identity];
Show[fd,c[1],c[2],c[3],c[4],c[5],c[6],
    DisplayFunction->$DisplayFunction];
```

**Problem:** Animate the phase portrait associated with the linear ODE system  $x' = \pi x - 2y$ ,  $y' = 4x - y$ ,  $x(0) = 1/2C$ ,  $y(0) = 1/2C$ , where  $C \in [-1, 1]$  (use the *Maple* functions `animate` and `phaseportrait`).

*Maple:*

```
with(plots): with(DEtools):
Vars:=[x(t),y(t)];
Eqs:=[(D(x))(t)=Pi*x(t)-2*y(t),(D(y))(t)=4*x(t)-y(t)];
ICs:=[[x(0)=1/2*C,y(0)=1/2*C]];
animate(phaseportrait,
[Eqs,Vars,t=0..Pi,ICs,x=-20..20,y=-20..20,
    stepsize=0.1,scaling=constrained,linecolor=blue],
    C=-1..1);
```

### 10.3 Partial Differential Equations

*Analytical solutions* to partial differential equations.

*Maple:*

- `pdsolve`, find analytical solutions for a given partial differential equation PDE and systems of PDEs;
- `declare`, declaring functions and derivatives on the screen for a simple, compact display;
- `casesplit`, split into cases and sequentially decouple a system of differential equations;
- `separability`, determine under what conditions it is possible to obtain a complete solution through separation of variables, etc.

```
with(PDEtools); pdsolve(PDE,f,HINT,build);
pdsolve({PDEs},{fs},HINT,ops); declare({funcs}; ON;
casesplit({PDEs},ops); separability(PDE,f,ops);
```

where `HINT=arg` are some hints, with `build` can be constructed an explicit expression for the indeterminate function `f`.

*Mathematica:*

- `DSolve`, find analytical solutions for a partial differential equation PDE and systems of PDEs.

```
DSolve[PDE,u[x1,...,xn],{x1,...,xn}]
DSolve[PDE,u,{x1,...,xn}]
```

**Problem:** Find a general solution to the wave equation  $u_{tt} = c^2 u_{xx}$ .

*Maple:*

```
with(PDEtools); declare(u(x,t)); ON;
pde:=diff(u(x,t),t$2)=c^2*diff(u(x,t),x$2); casesplit(pde);
separability(pde,u(x,t)); pdsolve(pde,build);
```

*Mathematica:*

```
pde1=D[u[x,t],{t,2}]-c^2*D[u[x,t],{x,2}]==0
DSolve[pde1,u[x,t],{x,t}]/.FullSimplify
pde2=(D[#, {x1,2}]-c^2*D[#, {x2,2}])&[y[x1,x2]]==0
DSolve[pde2,y[x1,x2],{x1,x2}]
```

**Problem:** Solve the initial-value problem  $u_t + t^2 u_x = 9$ ,  $u(x, 0) = x$  using the *method of characteristics*. Graph the characteristics.

*Maple:*

```
with(plots);
ODE:=diff(U(t),t)=9; Sol_Ch:=dsolve({ODE,U(0)=X[0]});
Eq_Ch:=diff(x(t),t)=t^2*U(t); Eq_Ch:=subs(Sol_Ch,Eq_Ch);
Cur_Ch:=dsolve({Eq_Ch, x(0)=X[0]}); Rt:=0..4; Rx:=-40..40;
display([seq(plot([subs(X[0]=x,eval(x(t),Cur_Ch)),t,t=Rt],
color=blue,thickness=2),x=Rx)],view=[Rx,Rt]);
u := unapply(subs(X[0]=solve(subs(x(t)=x,Cur_Ch),X[0]),
eval(U(t),Sol_Ch)),x,t);
```

*Mathematica:*

```
solCh=DSolve[{U'[t]==9,U[0]==X[0]},U[t],t]
eqCh=x'[t]==t^2*U[t]/.solCh[[1]]
curCh=DSolve[{eqCh,x[0]==X[0]},x[t],t]/.FullSimplify
g=Table[Plot[{curCh[[1,1,2]]/.X[0]->x},{t,0,4},
PlotStyle->Hue[0.7],PlotRange->{-40,40},
AspectRatio->1,DisplayFunction->Identity],
{x,-40,40}];
Show[g,DisplayFunction->$DisplayFunction];
uu=solCh[[1]]/.Solve[curCh[[1,1,2]]==x,X[0]]
u[X_,T_]:=uu[[1,1,2]]/.{x->X,t->T}; u[X,T]
```

*Numerical and graphical solutions to PDEs.*

*Maple:*

`pdsolve[numeric]`, find numerical solutions to a partial differential equation PDE or a system of PDEs.

Note that the solution obtained is represented as a module (similar to a procedure, see Subsect. 1.3.4) which can be used for obtaining visualizations (`plot`, `plot3d`, `animate`, `animate3d`) and numerical values (`value`), in more detail, see `?pdsolve[numeric]`.

```
with(PDEtools):
pdsolve({PDEs},{ICsBCs},numeric,{vars},ops);
Sol:=pdsolve({PDEs},{ICsBCs},numeric,{vars},ops);
Sol := animate(var, t=t0..t1, x=x0..x1, ops);
Sol := plot3d(var, t=t0..t1, ops);
Num_vals := Sol :- value(); Num_vals(num1, num2);
```

where `ICsBCs` are initial and boundary conditions, `vars` or `var` are dependent variables. It should be noted that `pdsolve[numeric]` returns a module with a number of solution forms (`plot`, `plot3d`, `animate`, `value`), and in this case the operator `:-` is used.

*Mathematica:*

`NDSolve`, find numerical solutions to PDE or a system of PDEs.

```
NDSolve[PDE,u,{x,x1,x2},{t,t1,t2},...]
NDSolve[PDE,{u1,...,un},{x,x1,x2},{t,t1,t2},...]
```

**Problem:** Find numerical and graphical solutions to the boundary value problem for the wave equation,

$$\begin{aligned} u_{tt} &= 0.01 u_{xx}, \quad u(0, t) = 0, \quad u(1, t) = 0, \\ u(x, 0) &= 0, \quad u_t(x, 0) = \sin(2\pi x), \end{aligned}$$

in the domain  $\mathcal{D} = \{0 < x < 1, 0 < t < \infty\}$ .

*Maple:*

```
with(VectorCalculus): with(plots): with(PDEtools): C:=0.01:
Eq:=diff(u(x,t),t$2)-C*Laplacian(u(x,t),'cartesian'[x])=0;
BC:= {u(0,t)=0, u(1,t)=0};
IC:= {D[2](u)(x,0)=sin(2*Pi*x), u(x, 0)=0};
```

```

0ps:=spacestep=1/100, timestep=1/100;
Sol:= pdsolve({Eq}, IC union BC, numeric, u(x,t), 0ps);
Sol :- animate(u(x,t),t=0..5*Pi,x=0..1,
               frames=30, numpoints=100,thickness=3,color=blue);
Sol :- plot3d(u(x,t), t=0..5*Pi, shading=zhue, axes=boxed);
Num_vals := Sol :- value(); Num_vals(1/2, Pi);

```

*Mathematica:*

```

<<Graphics`Animation`;
BCs={u[0,t]==0,u[1,t]==0};
pde={D[u[x,t],{t,2}]-0.01*D[u[x,t],{x,2}]==0};
ICs={(D[u[x,t],t]/.{t->0})==Sin[2*Pi*x],u[x,0]==0}
sol=NDSolve[{pde,ICs,BCs},u,{x,0,1},{t,0,5*Pi}];
f=u/.sol[[1]];
Plot3D[f[x,t],{x,0,1},{t,0,5*Pi}];
PaddedForm[Table[{x,f[x,Pi]},{x,0,1,0.1}]
           //TableForm,{12,5}]
Animate[Plot[f[x,t],{t,0,5*Pi},
            PlotStyle->{Hue[0.7],Thickness[0.02]}],{x,0,1},
        PlotRange->{{0,5*Pi},{-Pi/2,Pi/2}}];

```

**Problem:** Apply the explicit finite difference method for solving the initial boundary-value problem describing the movement of a fixed string

$$u_{tt} = \frac{1}{16\pi^2} u_{xx}, \quad 0 < x < 0.5, \quad t > 0, \quad u(0, t) = 0, \quad u(0.5, t) = 0, \\ u(x, 0) = 0, \quad u_t(x, 0) = \sin(4\pi x).$$

*Maple:*

```

L:=0.5; c:=1/(4*Pi); T:=0.5; m:=40; n:=40;
F:=X->0; G:=X->sin(4*Pi*X);
m1:=m+1; m2:=m-1; n1:=n+1; n2:=n-1;
h:=L/m; k:=T/n; lambda:=evalf(c*k/h);
for j from 2 to n1 do
  w[0,j-1]:=0; w[m1-1,j-1]:=0; od;
w[0,0]:=evalf(F(0)); w[m1-1,0]:=evalf(F(L));
for i from 2 to m do
  w[i-1,0]:=F(h*(i-1));
  w[i-1,1]:=(1-lambda^2)*F(h*(i-1))
    +lambda^2*(F(i*h)+F(h*(i-2)))/2+k*G(h*(i-1));
od;

```

```

for j from 2 to n do
  for i from 2 to m do
    w[i-1,j] := evalf(2*(1-lambda^2)*w[i-1,j-1]
      +lambda^2*(w[i,j-1]+w[i-2,j-1])-w[i-1,j-2]);
  od;
od;
printf(' i X(i) w(X(i),n)\n';
for i from 1 to m1 do
  X[i-1] := (i-1)*h:
  printf('%3d %11.8f %13.8f\n',i,X[i-1],w[i-1,n1-1]);
od:
Points:=[seq([X[i-1], w[i-1, n1-1]], i=1..m1)];
plot(Points,style=point,color=blue,symbol=circle);

```

*Mathematica:*

```

{l=0.5, c=1/(4*Pi)//N, t=0.5, m=41, n=41,
 h=1/(m-1), k=t/(n-1), \[Lambda]=c*k/h//N}
f[x_]:=0; g[x_]:=Sin[4*Pi*x];
fi[i_]:=f[h*(i-1)]; gi[j_]:=g[k*(j-1)];
w=Table[0,{n},{m}];
For[i=1,i<=n,i++, w[[i,1]]=fi[i]];
For[i=2,i<=n-1,i++,
  w[[i,2]]=(1-\[Lambda]^2)*fi[i]
  +\[Lambda]^2*(fi[i+1]+fi[i-1])/2
  +k*gi[i];
For[j=3,j<=m,j++,
  For[i=2,i<=n-1,i++,
    w[[i,j]]=2*(1-\[Lambda]^2)*w[[i,j-1]]
    +\[Lambda]^2*(w[[i+1,j-1]]+w[[i-1,j-1]])
    -w[[i,j-2]]//N];
Print[" i x(i) w(x(i),n)", "\n"];
For[i=1,i<=n,i++,
  Print[PaddedForm[i,2],
    PaddedForm[h*(i-1),7], " ",
    PaddedForm[w[[i,n]],10]]];
points=Table[{h*(i-1),w[[i,n]]},{i,1,n}]
ListPlot[points,PlotStyle->\{Blue,PointSize[0.02]\}];
Print[NumberForm[TableForm[Transpose[Chop[w]]],5]];
ListPlot3D[w,ViewPoint->\{3,1,3\},ColorFunction->Hue];

```

*Application to nonlinear standing waves* in a fluid using the Lagrangian formulation.

**Problem:** Construct approximate analytical solutions describing nonlinear standing waves on the free surface of a fluid.

*Statement of the problem.* The classical two-dimensional standing waves problem consists of solving the Euler equations for a one- or two-layer fluid with boundary conditions. The assumption is made that the flow is irrotational. The boundary value problem needs to be solved in a flow domain,  $\mathcal{D} = \{0 \leq x \leq L, -h \leq y \leq \eta(x, t)\}$  for the surface elevation  $\eta(x, t)$  and the velocity potential  $\phi(x, y, t)$ . The fluid depth  $h$ , the surface tension constant  $T$ , and the horizontal size of the domain  $L$  are given. We study periodic solutions (in  $x$  and  $t$ ) of a standing waves problem.

In general, there are two ways for representing the fluid motion: the Eulerian approach, in which the coordinates are fixed in the reference frame of the observer, and the Lagrangian approach, in which the coordinates are fixed in the reference frame of the moving fluid.

Approximate solutions in Lagrangian variables are constructed analytically. The analytic Lagrangian approach proposed by [31] is followed for constructing approximate solutions for the nonlinear waves. We generalize the solution method to allows us to solve a set of problems, for example, the problems of infinite- and finite-depth surface standing waves and infinite- and finite-depth internal standing waves. This method can be useful for extending a series solutions to higher order, solving a problem that is not solvable in Eulerian formulation, or to solve other set of problems. We develop computer algebra procedures to aid in the construction of higher-order approximate analytical solutions.

Most of the approximate analytic solutions have been obtained using the Eulerian formulation, this section deals with the alternative formulation, which deserves to be considered. Therefore, we compare the analytic frequency-amplitude dependences obtained in Lagrangian variables with the corresponding ones known in Eulerian variables. The analysis has shown that the analytic frequency-amplitude dependences are in complete agreement with previous results obtained by [28], [24], [5], [37], [23] in Eulerian variables, and by [32], [33], [35] in Lagrangian variables.

The analysis of solutions has shown that the use of the Lagrangian approach to solve standing waves problems presents some advantages

with respect to the Eulerian formulation, particularly because it allows us to simplify the boundary conditions (the unknown free boundary is a line), the radius of convergence of an expansion parameter is larger than in the Eulerian variables (this allows us to observe steep standing waves).

*Asymptotic solution.* Let us consider two-dimensional nonlinear wave motions in the fluid domain  $\mathcal{D} = \{0 \leq x \leq L, -\infty \leq y \leq \eta(x, t)\}$ . On the free surface the pressure is constant and equal to zero. We consider this as a basic model and other models can be derived from this one.

The rectangular system of coordinates  $xOy$  in the plane of motion is chosen so that the  $x$ -axis coincides with the horizontal level of fluid at rest and the  $y$ -axis is directed vertically upwards so that the unperturbed free surface has coordinates  $y = 0$  and  $x \in [0, L]$ .

The transformation of variables from Eulerian  $(x, y)$  to Lagrangian  $(a, b)$  are carried out adding the following requirements:

the Jacobian  $J = \partial(x, y)/\partial(a, b) = 1$ ,

the free surface  $y = \eta(x, t)$  is equivalent to the parametric curve  $\{x(a, 0, t), y(a, 0, t)\}$ ,

at  $t = 0$  the free surface is  $\{x(a, 0, 0), y(a, 0, 0)\}$ ,

at the vertical lines  $a = 0$  and  $a = L$  the horizontal velocity  $x_t = 0$ ,

the infinite depth is  $b = -\infty$ .

The Lagrange equations for wave motions in fluid, the continuity equation and the boundary conditions are then given as:

$$x_{tt}x_a + (y_{tt} + g)y_a + \frac{p_a}{\rho} = 0, \quad x_{tt}x_b + (y_{tt} + g)y_b + \frac{p_b}{\rho} = 0,$$

$$\frac{\partial(x, y)}{\partial(a, b)} = 1,$$

$$x(0, b, t) = 0, \quad x(L, b, t) = L, \quad y(a, -\infty, t) = -\infty, \quad p(a, 0, t) = p_T,$$

where  $x(a, b, t)$  and  $y(a, b, t)$  are the coordinates of an individual fluid particle in motion,  $p(a, 0, t)$  is the pressure on the free surface due to surface tension,  $\rho$  is the fluid density. We study a weak capillarity and include surface tension similar to the parametric expression (2.7) written

by Schultz et al. [36]. In Lagrangian coordinates  $p_T$  can be computed by means of

$$p_T = -T \left[ \frac{y_{aa}x_a - y_a x_{aa}}{(x_a^2 + y_a^2)^{3/2}} \right], \quad (10.1)$$

where  $T$  is the surface tension constant. The dispersion relation for linear waves is  $\omega_{(0)}^2 = g\kappa(1 + T_z)$ , where the dimensionless surface tension is  $T_z = \kappa^2 T / (\rho g)$ .

Note that we follow the notation of Concus [9],  $\delta = \frac{T_z}{1 + T_z}$ , for more compact presentation of the result.

Let us consider weakly nonlinear standing waves or waves of small amplitude and steepness, for which the amplitude and the ratio of wave height to wavelength is assumed to be of order  $\varepsilon$ , where  $\varepsilon$  is a small parameter.

We introduce the dimensionless amplitude  $\varepsilon$ , the wave phase  $\psi$ , Lagrangian variables  $\alpha, \beta$  (instead of  $a, b$ ), and space coordinates and pressure  $\xi, \eta$ , and  $\sigma$  (instead of  $x, y$ , and  $p$ ):

$$\begin{aligned} \kappa A &= \varepsilon, & \psi &= \omega t, & \alpha &= a\kappa, & \beta &= b\kappa, \\ \kappa x &= \alpha + \varepsilon\xi, & \kappa y &= \beta + \varepsilon\eta, & \kappa^2 p &= -\kappa(\rho g)\kappa y + \varepsilon\rho\omega_{(0)}^2\sigma, \end{aligned}$$

where  $\kappa = \pi n/L$  is the wave number ( $n$  is the number of nodes of the wave),  $\omega$  is the nonlinear frequency,  $\omega_{(0)}^2 = g\kappa$  is the dispersion relation for linear periodic waves, and  $g$  is the acceleration due to gravity.

In terms of the dimensionless variables, the equations of motion and the boundary conditions can be rewritten in the form

$$\begin{aligned} \mathcal{L}^1(\xi, \sigma) &= -\varepsilon(\xi_{\psi\psi}\xi_\alpha + \eta_{\psi\psi}\eta_\alpha), & \mathcal{L}^2(\eta, \sigma) &= -\varepsilon(\xi_{\psi\psi}\xi_\beta + \eta_{\psi\psi}\eta_\beta), \\ \mathcal{L}^3(\xi, \eta) &= -\varepsilon \frac{\partial(\xi, \eta)}{\partial(\alpha, \beta)}, \\ \xi(0, \beta, \psi) &= 0, & \xi(\pi n, \beta, \psi) &= 0, & \eta(\alpha, -\infty, \psi) &= 0, \\ \sigma(\alpha, 0, \psi) - \eta(\alpha, 0, \psi) &= -T_z \varepsilon \kappa \left[ \frac{\eta_{\alpha\alpha} + \varepsilon(\eta_{\alpha\alpha}\xi_\alpha - \eta_\alpha\xi_{\alpha\alpha})}{[1 + 2\varepsilon\xi_\alpha + \varepsilon^2(\xi_\alpha^2 + \eta_\alpha^2)]^{3/2}} \right], \end{aligned}$$

where linear differential operators  $\mathcal{L}^i$  ( $i = 1, 3$ ) are

$$\mathcal{L}^1(\xi, \sigma) = \xi_{\psi\psi} + \sigma_\alpha, \quad \mathcal{L}^2(\eta, \sigma) = \eta_{\psi\psi} + \sigma_\beta, \quad \mathcal{L}^3(\xi, \eta) = \xi_\alpha + \eta_\beta.$$

The construction of asymptotic solutions is based on perturbation theory. Defining the formal power series in the amplitude parameter  $\varepsilon$

$$u = \mathcal{F}^{111}(u) + \sum_{i=2}^N \varepsilon^{i-1} u^{(i)} + O(\varepsilon^N), \quad u = \xi, \eta, \sigma,$$

where  $\xi^{(i)}, \eta^{(i)}$ , and  $\sigma^{(i)}$  ( $i = 2, \dots, N$ ) are unknown  $2\pi$ -periodic in  $\psi$  functions of variables  $\alpha$ ,  $\beta$ , and  $\psi$ , and the linear terms  $\mathcal{F}^{111}(u)$ ,  $u = \xi, \eta, \sigma$ , are defined by the following expressions:

$$\mathcal{F}^{111}(\xi) = -\sin(\alpha)e^\beta \cos(\psi), \quad \mathcal{F}^{111}(\eta) = \mathcal{F}^{111}(\sigma) = \cos(\alpha)e^\beta \cos(\psi).$$

Considering weakly nonlinear standing waves, we can assume that the nonlinear wave frequency  $\omega$  is close to the linear wave frequency  $\omega_{(0)}$ :

$$\omega(\varepsilon) \equiv \psi_t = \omega_{(0)} + \sum_{i=1}^{N-1} \varepsilon^i \omega_{(i)} + O(\varepsilon^N),$$

where  $\omega_{(i)}$  are new unknown corrections to the nonlinear wave frequency.

Substituting these expansions into the equations of motion and the boundary conditions and matching the coefficients of like powers of  $\varepsilon$ , we arrive at the following linear inhomogeneous system of partial differential equations and boundary conditions for the  $i$ -th approximation:

$$\begin{aligned} \mathcal{L}^1(\xi^{(i)}, \sigma^{(i)}) &= \mathcal{S}^1(\xi^{(i)}), & \mathcal{L}^2(\eta^{(i)}, \sigma^{(i)}) &= \mathcal{S}^2(\eta^{(i)}), \\ \mathcal{L}^3(\xi^{(i)}, \eta^{(i)}) &= \mathcal{S}^3(\sigma^{(i)}), \\ \xi^{(i)}(0, \beta, \psi) &= 0, & \xi^{(i)}(\pi n, \beta, \psi) &= 0, & \eta^{(i)}(\alpha, -\infty, \psi) &= 0, \\ \sigma^{(i)}(\alpha, 0, \psi) - \eta^{(i)}(\alpha, 0, \psi) &= -T_z \varepsilon \kappa \mathcal{E}^{(i)}, \end{aligned} \tag{10.2}$$

where

$$\begin{aligned} \mathcal{E}^{(i)} &= \left[ \frac{\eta_{\alpha\alpha}^{(i)} + \varepsilon(\eta_{\alpha\alpha}^{(i)} \xi_\alpha^{(i)} - \eta_\alpha^{(i)} \xi_{\alpha\alpha}^{(i)})}{[1 + 2\varepsilon \xi_\alpha^{(i)} + \varepsilon^2((\xi_\alpha^{(i)})^2 + (\eta_\alpha^{(i)})^2)]^{3/2}} \right], \\ \mathcal{S}^m(u^{(i)}) &= \sum_{j,k,l=0}^i [F_{mi}^{jkl} \mathcal{F}^{jkl}(u^{(i)}) + G_{mi}^{jkl} \mathcal{F}_\psi^{jkl}(u^{(i)})], \quad u^{(i)} = \xi^{(i)}, \eta^{(i)}, \sigma^{(i)}, \\ \mathcal{F}^{ijk}(\xi^{(i)}) &= \sin(i\alpha)e^{(j\beta)} \cos(k\psi), \\ \mathcal{F}^{ijk}(\eta^{(i)}) &= \mathcal{F}^{ijk}(\sigma^{(i)}) = \cos(i\alpha)e^{(j\beta)} \cos(k\psi). \end{aligned}$$

We omit the expressions for all coefficients  $F_{mi}^{jkl}, G_{mi}^{jkl}$  ( $m=1, 2, 3, j, k, l=0, \dots, i, i=2, \dots, N$ ) because of their great complexity.

We look for a solution,  $2\pi$ -periodic in  $\psi$ , to this system of equations and the boundary conditions in the form:

$$v^{(i)} = \sum_{j,k,l=0}^i V_i^{jkl} \mathcal{F}^{jkl}(v), \quad v = \xi, \eta, \sigma, \quad V = \Xi, H, \Sigma,$$

where  $\Xi_i^{jkl}, H_i^{jkl}$ , and  $\Sigma_i^{jkl}$  are unknown constants.

By using formulas described above, we obtain the asymptotic solution of the order of  $O(\varepsilon^N)$  and the unknown corrections to the nonlinear wave frequency  $\omega_{(i)}$ . Setting  $\beta = 0$  in the parametric equations for  $\kappa x$  and  $\kappa y$ , we can obtain the profiles of surface standing waves in Lagrangian variables for the  $i$ -th approximation ( $i = 1, \dots, N$ ). Changes in the amplitude  $\varepsilon$  influence the surface configuration by changing both the shape of the surface and the amplitude of motion.

*Approximate analytical solution with Maple.* We present the *Maple* solution for every stage of the method and up to the second approximation,  $N = 2$  (NA=2).

(1) We find the second derivative with respect to dimensionless time  $\psi$  for the variables  $\xi(\alpha, \beta, t)$  and  $\eta(\alpha, \beta, t)$ :

```
NA := 2; NP:=NA-1; NN:=NA+1; TZ :=-delta/(delta-1); lambda:=1+TZ;
psiT := omega+add(epsilon^i*omega||i,i=1..NP);
setsub:={diff(psi(t),t)=psiT}; subt:={psi(t)=psi};
xi := (x,y,z)->-sin(x)*exp(y)*cos(z);
eta := (x,y,z)-> cos(x)*exp(y)*cos(z);
sigma:=(x,y,z)-> cos(x)*exp(y)*cos(z);
Fxi := (x-> xi(alpha,beta,psi(x)));
Feta := (x->eta(alpha,beta,psi(x)));
dxi := diff(Fxi(t),t);      deta := diff(Feta(t),t);
xi1T := subs(setsub,dxi);   xi2T := subs(setsub,diff(xi1T,t));
eta1T:=subs(setsub,deta);  eta2T:=subs(setsub,diff(eta1T,t));
Xi:=xi(alpha,beta,psi);    Eta:=eta(alpha,beta,psi);
Sigma:=sigma(alpha,beta,psi);
```

It should be noted that, for convenience, we exclude the symbolic expressions of linear operators  $\mathcal{L}^1$ ,  $\mathcal{L}^2$ , and  $\mathcal{L}^3$  in the time derivatives and the governing equations. We are working with the right-hand sides

(trigonometric parts) of the expressions in (10.2). For example, in the second time derivative

$$\begin{aligned}\frac{\partial^2 \eta^{(2)}}{\partial t^2} &= -\omega^2 \cos \alpha^2 \exp \beta \cos \psi \\ &\quad + \varepsilon \left( \omega^2 \frac{\partial^2 \eta^{(2)}}{\partial \psi^2} - 2\omega \cos \alpha^2 \exp \beta \cos \psi \right) + O(\varepsilon^2),\end{aligned}$$

and in  $\mathcal{L}^2$ , the second equation in (10.2), we omit the symbolic expression  $\eta_{\psi\psi}^{(2)}$  (the second approximation,  $\text{NA}=2$ ).

(2) The equations of motion and the continuity equation:

```
F1:=-omega^2*diff(Sigma,alpha)-epsilon*(xi2T*diff(Xi,alpha)
+eta2T*diff(Eta, alpha))-xi2T;
F2:=-omega^2*diff(Sigma,beta)-epsilon*(xi2T*diff(Xi,beta)
+eta2T*diff(Eta, beta))-eta2T;
F3:=-diff(Xi,alpha)-diff(Eta,beta)+epsilon*(diff(Xi,beta)
*diff(Eta,alpha)-diff(Eta,beta)*diff(Xi,alpha));
for i from 1 to 3 do
F||i||S:=coeff(subs(subt,F||i),epsilon,NP)*epsilon^NP:
od;
```

(3) The boundary conditions:

```
Bc1:=xi=evala(subs(alpha=0,coeff(xi(alpha,beta,psi),
epsilon,NP)));
Bc2:=xi=evala(subs(alpha=Pi*n,coeff(xi(alpha,beta,psi),
epsilon,NP)));
Bc3:=eta=evala(subs(beta=-infinity,
coeff(eta(alpha,beta,psi),epsilon,NP)));
Bc4:=(x, y)->lambda*x-y+TZ*diff(y,alpha$2)
+TZ*(S-diff(y,alpha$2));
Bc4_1:=Bc4(sigma(alpha,beta,psi),eta(alpha,beta,psi));
S_numer:=((1+epsilon*diff(xi(alpha,beta,psi),alpha))
*diff(eta(alpha,beta,psi),alpha$2)-epsilon
*diff(xi(alpha,beta,psi),alpha$2)
*diff(eta(alpha,beta,psi),alpha));
S_denom:=(1+epsilon*diff(xi(alpha,beta,psi),alpha))^2
+(epsilon*diff(eta(alpha,beta,psi),alpha))^2;
S:=convert(series(S_numer/S_denom^(3/2),epsilon,NA),polynom);
Bc4_2:=evala(subs(beta=0,Bc4_1));
F4S :=coeff(Bc4_2,epsilon, NP)*epsilon^NP;
```

(4) We rewrite the governing equations describing the standing wave motion and the fourth boundary condition at the free surface in the form of *Maple* functions:

```

Eq1:=(N,M,K,L)->-K^2*(Xi||N||M||K||L)-N*(Sigma||N||M||K||L)
      =combine(EqC(F1S,N,M,K)/epsilon^NP);
Eq2:=(N,M,K,L)->-K^2*(Eta||N||M||K||L)+M*(Sigma||N||M||K||L)
      =combine(EqC(F2S,N,M,K)/epsilon^NP);
Eq3:=(N,M,K,L)->N*(Xi||N||M||K||L)+M*(Eta||N||M||K||L)
      =combine(EqC(F3S,N,M,K)/epsilon^NP);
Bc4:=(N,M,K,L)->lambda*(Sigma||N||M||K||L)-(Eta||N||M||K||L)
      -N^2*TZ*(Eta||N||M||K||L);

```

(5) The coefficients  $F_{mi}^{jkl}$  and  $G_{mi}^{jkl}$  ( $m = 1, 2, 3$ ,  $j, k, l = 0, \dots, i$ ) can be calculated according to the orthogonality conditions. On the basis of the orthogonality property of eigenfunctions, we create the procedures S1\_NM, S2\_NM, and Bc4\_NK, which are the functions in the procedures EqC and BcC:

```

S1_NM:=proc(x,N,M,K,OM2)
  local i,I_N,I_M,I_K,T,TI,CTI,Z3,NZ3,TII,CTII,TT,I_T;
  I_N:=factor(combine(1/Pi*int(combine(x,exp)*sin(N*alpha),
    alpha=0..2*Pi),exp,trig));
  if K=0 then I_K:=1/(2*Pi)*int(I_N*cos(K*psi),psi=0..2*Pi);
  else I_K:=1/Pi*int(I_N*cos(K*psi),psi=0..2*Pi); fi:
  I_M:=0: Z3:=collect(I_K,[exp,delta]):
  if Z3 = 0 then I_T := 0: else
    if type(Z3, '+')=true then NZ3:=nops(Z3):
      for i from 1 to NZ3 do
        TI[i] :=select(has,op(i,Z3),beta);
        CTI[i] :=remove(has,op(i,Z3),beta);
        if TI[i]=0 then T[i]:=0:
        else T[i]:=evala(subs(beta=0,diff(TI[i],beta))); fi:
        if T[i] = 0 or T[i] <> M then I_M := I_M + 0;
        else I_M := I_M + CTI[i]/OM2; fi:
        od: I_T:=I_M:
    else TII:=select(has,Z3,beta); CTII:=remove(has,Z3,beta);
      if TII = 0 then TT := 0:
      else TT := evala(subs(beta=0, diff(TII, beta))); fi:
      if TT = 0 or TT <> M then I_M := I_M + 0;
      else I_M := I_M + CTII; fi: I_T:=I_M/OM2:
    fi: fi: RETURN(I_T); end;

```

```

S2_NMk:=proc(x,N,M,K,OM2)
    local i,I_N,I_M,I_K,T,TI,CTI,Z3,NZ3,TII,CTII,TT,I_T;
    if N=0 then I_N:=1/(2*Pi)*int(x*cos(N*alpha),alpha=0..2*Pi);
    else I_N:=1/Pi*int(x*cos(N*alpha),alpha=0..2*Pi); fi:
    if K=0 then I_K:=1/(2*Pi)*int(I_N*cos(K*psi),psi=0..2*Pi);
    else I_K:=1/Pi*int(I_N*cos(K*psi),psi=0..2*Pi); fi:
    I_M:=0: Z3:=expand(I_K,exp,sin,cos):
    if Z3=0 then I_T:=0: else
        if type(Z3, '+')=true then NZ3:=nops(Z3):
            for i from 1 to NZ3 do
                TI[i] :=select(has,op(i,Z3),beta);
                CTI[i] :=remove(has,op(i,Z3),beta);
                if TI[i]=0 then T[i]:=0:
                    else T[i]:=evala(subs(beta=0,diff(TI[i],beta))); fi:
                if T[i]=0 or T[i] <> M then I_M:=I_M+0;
                else I_M := I_M + CTI[i]/OM2; fi:
            od: I_T:=I_M:
        else TII:=select(has,Z3,beta); CTII:=remove(has,Z3,beta);
            if TII=0 then TT:=0:
                else TT:=evala(subs(beta=0,diff(TII,beta))); fi:
            if TT=0 or TT <> M then I_M:=I_M+0;
            else I_M:=I_M+CTII; fi: I_T:=I_M/OM2:
        fi: fi: RETURN(I_T); end;

Bc4_NK := proc(x, N, K) local I_T;
    I_T := 1/Pi^2*int(int(x*cos(N*alpha)*cos(K*psi),
                           alpha=0..2*Pi), psi=0..2*Pi);
    if N=0 then I_T:=I_T/2 fi; if K=0 then I_T:=I_T/2 fi;
RETURN(I_T); end;

```

(6) To find the coefficients in the governing equations and the fourth boundary condition, we create the procedures EqC and BcC:

```

EqC := proc(Eq, N, M, K) local SS;      SS:=0:
    if Eq=F1S then SS:=S1_NMk(Eq,N,M,K,omega^2):
    elif Eq=F2S then SS:=S2_NMk(Eq,N,M,K,omega^2):
    elif Eq=F3S then SS:=S2_NMk(Eq,N,M,K,1): fi:
RETURN(SS) end;
BcC := proc(Eq, N, K) local SS,NEq,i; SS:=0:
    if type(Eq, '+')=false then SS:=SS+Bc4_NK(Eq,N,K):
    else NEq:=nops(Eq);
        for i from 1 to NEq do SS:=SS+Bc4_NK(op(i,Eq),N,K): od: fi:
RETURN(SS) end;

```

(7) We solve these systems and obtain the asymptotic solution of the order of  $O(\varepsilon^N)$ , using the equations described above:

```

system1 := {Eq1(2,0,0,2), Eq2(2,0,0,2), Eq3(2,0,0,2),
           Eq1(2,2,0,2), Eq2(2,2,0,2), Eq3(2,2,0,2),
           Bc4(2,0,0,2)+Bc4(2,2,0,2)=simplify(BcC(F4S,2,0)/epsilon^NP)};
system1 := system1 union {Eta200||NA=0, Eta200||NA=0};
var1 := {Xi200||NA, Eta200||NA, Sigma200||NA,
          Xi220||NA, Eta220||NA, Sigma220||NA};
DF1 := solve(system1, var1);
system2 := {Eq1(2,0,2,2), Eq2(2,0,2,2), Eq3(2,0,2,2),
           Eq1(2,2,2,2), Eq2(2,2,2,2), Eq3(2,2,2,2),
           Bc4(2,0,2,2)+Bc4(2,2,2,2)=simplify(BcC(F4S,2,2)/epsilon^NP)};
system2 := system2 union {Eta202||NA=0, Eta202||NA=0};
var2 := {Xi202||NA, Eta202||NA, Sigma202||NA,
          Xi222||NA, Eta222||NA, Sigma222||NA};
DF2 := solve(system2, var2);
system3 := {Eq1(0,2,0,2), Eq2(0,2,0,2), Eq3(0,2,0,2),
           Eq1(0,0,0,2), Eq2(0,0,0,2), Eq3(0,0,0,2),
           Bc4(0,2,0,2)+Bc4(0,0,0,2)=simplify(BcC(F4S,0,0)/epsilon^NP)};
system3 := system3 union
           {Xi020||NA=0, Xi000||NA=0, Sigma000||NA=0};
var3 := {Xi020||NA, Eta020||NA, Sigma020||NA,
          Xi000||NA, Eta000||NA, Sigma000||NA};
DF3 := solve(system3, var3);
system4 := {Eq1(0,2,2,2), Eq2(0,2,2,2), Eq3(0,2,2,2),
           Eq1(0,0,2,2), Eq2(0,0,2,2), Eq3(0,0,2,2),
           Bc4(0,2,2,2)+Bc4(0,0,2,2)=simplify(BcC(F4S,0,2)/epsilon^NP)};
system4 := system4 union {Eta002 || NA = 0};
var4 := {Xi022||NA, Eta022||NA, Sigma022||NA,
          Xi002||NA, Eta002||NA, Sigma002||NA};
DF4 := solve(system4, var4);
system5 := {Eq1(1,1,1,2), Eq2(1,1,1,2), Eq3(1,1,1,2),
           Bc4(1,1,1,2)=simplify(BcC(F4S,1,1)/epsilon^NP)};
system5 := system5 union {Sigma111||NA=0};
var5 := {Xi111||NA, Eta111||NA, Sigma111||NA, omega||NP};
DF5S := solve(system5, var5);
for i from 1 to nops(DF5S) do
  if has(op(i,DF5S), omega||NP) then
    ZZ:=i: oomega1:=op(i,DF5S): fi: od:
  DF5 := subsop(ZZ=NULL,DF5S);

```

(8) We generate the approximate analytical solution for the second approximation:

```
FF:=proc(x::list) local Xi_S,Eta_S,Sigma_S,i,Nx,a,b,c,s,N,M,K;
Nx:=nops(x); Xi_S:={} : Eta_S:={} : Sigma_S:={} :
for i from 1 to Nx do a:=lhs(x[i]);
for N from 0 to NA do for M from 0 to NA do for K from 0 to NA do
if a=Xi||N||M||K||NA then
  CXi||N||M||K||NA:=rhs(x[i])*sin(N*alpha)*exp(M*beta)*cos(K*psi):
  Xi_S:=Xi_S union {CXi||N||M||K||NA};
elif a=Eta||N||M||K||NA then
  CEta||N||M||K||NA:=
    rhs(x[i])*cos(N*alpha)*exp(M*beta)*cos(K*psi):
  Eta_S:=Eta_S union {CEta||N||M||K||NA};
elif a=Sigma||N||M||K||NA then
  CSigma||N||M||K||NA:=
    rhs(x[i])*cos(N*alpha)*exp(M*beta)*cos(K*psi):
  Sigma_S:=Sigma_S union {CSigma||N||M||K||NA};
fi: od: od: od: od:
RETURN(Xi_S, Eta_S, Sigma_S); end;
ASet||NA:={op(DF||i)'$'i'=1..5};
BSet||NA:=convert(ASet||NA,list);
ASetf||NA:={} ; NS:=nops(BSet||NA);
for i from 1 to NS do
V||i:=op(i, BSet||NA):
if rhs(V||i)=0 then
  ASetf||NA:=ASetf||NA union {subs(rhs(V||i)=0,V||i)}:
else
  ASetf||NA:=ASetf||NA union {subs(rhs(V||i)=lhs(V||i),V||i)}:
fi: od:
BSetf||NA:=convert(ASetf||NA,list);
Xi||NA:=convert(FF(BSetf||NA)[1],'+');
Eta||NA:=convert(FF(BSetf||NA)[2],'+');
Sigma||NA:=convert(FF(BSetf||NA)[3],'+');
Xi_Ser :=xi(alpha,beta,psi)+epsilon^NN*Xi||NA;
Eta_Ser :=eta(alpha,beta,psi)+epsilon^NN*Eta||NA;
Sigma_Ser:=sigma(alpha,beta,psi)+epsilon^NN*Sigma||NA;
```

The case  $T=0$ . If surface tension is neglected, then the free boundary problem is well-posed in a finite interval of time if the sign condition  $-\nabla p \cdot \mathbf{n} > 0$  is satisfied [44]. Here  $p$  is the pressure and  $\mathbf{n}$  is the outward unit normal to the free boundary. Note that when the forces of surface

tension are taken into account, then the free boundary problem is always well-posed irrespective of the sign condition (see [3], [44]). This condition is somewhat similar to the fact that  $p > 0$  inside the water region. In our case, according to the solution obtained,  $\eta^{(i)}, \sigma^{(i)}$ , the pressure takes the form:

$$\begin{aligned} p(\alpha, \beta, t) = & -\frac{1}{12\kappa} \rho g (12\beta + \varepsilon^2 [-6e^{2\beta} \cos(2\psi) + 6 \cos(2\psi)] \\ & + \varepsilon^3 [-3 \cos(\alpha)e^{3\beta} \cos(\psi) + 3 \cos(\alpha)e^\beta \cos(\psi) \\ & - 5 \cos(\alpha)e^{3\beta} \cos(3\psi) + 5 \cos(\alpha)e^\beta \cos(3\psi)]) + O(\varepsilon^4). \end{aligned}$$

The parametrization of the free surface by Lagrangian coordinates  $(\alpha, \beta)$ , that is,  $\mathcal{B} = (x(\alpha, \beta, t), y(\alpha, \beta, t))$ , has the following form:

$$\begin{aligned} x(\alpha, \beta, t) = & -\frac{1}{96\kappa} (-96\alpha + 96\varepsilon \sin(\alpha)e^\beta \cos(\psi) \\ & + \varepsilon^3 [-5 \sin(\alpha)e^\beta \cos(3\psi) \\ & + 24 \sin(\alpha)e^\beta \cos(\psi)]) + O(\varepsilon^4), \\ y(\alpha, \beta, t) = & \frac{1}{96\kappa} (96\beta + 96\varepsilon \cos(\alpha)e^\beta \cos(\psi) \\ & + \varepsilon^2 [24e^{2\beta} + 24e^{2\beta} \cos(2\psi)] \\ & + \varepsilon^3 [24 \cos(\alpha)e^{3\beta} \cos(\psi) + 24 \cos(\alpha)e^\beta \cos(\psi) \\ & + 8 \cos(\alpha)e^{3\beta} \cos(3\psi) - 5 \cos(\alpha)e^\beta \cos(3\psi)]) + O(\varepsilon^4). \end{aligned}$$

It is easy to verify that the solution obtained satisfies the sign condition and the condition  $p > 0$  inside the water region.

*Frequency-amplitude dependence.* Applying the above method, we obtain the high-order asymptotic solution to the problem of capillary-gravity waves in an infinite-depth fluid. The analytic frequency-amplitude dependence is

$$\frac{\omega}{\omega_0} = 1 + \varepsilon^2 \left[ \frac{81\delta^3 + 36\delta^2 + 27\delta - 8}{64(3\delta+1)(1-3\delta)} \right] - \varepsilon^4 \left[ \frac{P^\infty}{16384 Q^\infty} \right] + O(\varepsilon^5),$$

where polynomials  $P^\infty$  and  $Q^\infty$  are:

$$\begin{aligned} P^\infty &= -16691184\delta^9 + 13314456\delta^8 - 876987\delta^7 - 726327\delta^6 + 3458214\delta^5 \\ &\quad - 3099546\delta^4 + 554373\delta^3 + 65721\delta^2 - 1216\delta - 1472, \\ Q^\infty &= (1+12\delta)(4\delta-1)(-1+3\delta)^3(3\delta+1)^2. \end{aligned}$$

This dependence coincides with the analytic result in Eulerian coordinates obtained by Concus [9] up to the 5-th order.

If surface tension is neglected,  $T = 0$ , applying the above method, we obtain the asymptotic solution to the problem of gravity waves in an infinite-depth fluid and we write out the frequency-amplitude dependence:

$$\frac{\omega}{\omega(0)} = 1 - \frac{1}{8}\varepsilon^2 - \frac{23}{256}\varepsilon^4 + O(\varepsilon^5).$$

This dependence is equal to the previous results obtained by [32], [35] in Lagrangian variables and equal to the analytic solution obtained by [28] in Eulerian variables, where  $\omega_1 = 0$  and  $\omega_2 = -\frac{1}{8}A^2\omega_0$ . This expression coincides with the results obtained by [24] and [5] in Eulerian variables, where  $\omega_3 = 0$  and  $\omega_4 = -\frac{15}{256}A^4\omega_0$ .

The coincidence of the results follows from determining the maximum amplitudes  $A_{\max}$  that corresponds to the maximum wave profiles  $y_{\max}$  in both types of variables and calculating the relation between them. In our case, after the passage from Lagrangian variables back to Eulerian, we have

$$\kappa y(x, t) = (\varepsilon + \frac{5}{32}\varepsilon^3) \cos \kappa x \cos t; \quad \kappa y_{\max} = (\varepsilon + \frac{5}{32}\varepsilon^3) \cos \kappa x, \quad (10.3)$$

where  $A_{\max} = \varepsilon + \frac{5}{32}\varepsilon^3$ . In the papers by [24] and [5] the maximum wave profile and the maximum amplitude are

$$y(x, t) = (A + \frac{1}{32}A^3) \cos x \sin \sigma t; \quad y_{\max} = (A + \frac{1}{32}A^3) \cos x, \quad (10.4)$$

where  $A_{\max} = A + \frac{1}{32}A^3$ . Equating the maximum amplitudes in both cases, we obtain the relation  $A = \varepsilon + \frac{1}{8}\varepsilon^3$ . Substituting the value of  $A$  into the frequency-amplitude dependence obtained by [24] and [5], we finally obtain

$$\frac{\omega}{\omega(0)} = 1 - \frac{1}{8}A^2 - \frac{15}{256}A^4 + O(A^5) = 1 - \frac{1}{8}\varepsilon^2 - \frac{23}{256}\varepsilon^4 + O(\varepsilon^5), \quad (10.5)$$

that is equal to the frequency-amplitude dependence obtained.

*Surface profiles.* We construct the standing wave profiles, using the approximate analytical solutions obtained upto the third approximation. According to the Lagrangian formulation, the free surface  $y = \eta(x, t)$  is defined by the parametric curve  $\{x(a, 0, t), y(a, 0, t)\}$ ,  $b=0$ . Setting  $b=0$  in the solutions obtained for  $x$  and  $y$ , we can observe the standing wave motion:

```
with(plots):
Digits:=30; Omega1:=0; Omega3:=0;
Omega2 := (81*delta^3+36*delta^2+27*delta-8)/
          (64*(3*delta+1)*(1-3*delta));
Param:=[T=72,g=981.7, n=2,L=50,rho=1];
kappa :=evalf(subs(Param,(Pi*n)/L));
TTZ:=evalf(subs(Param,T*kappa^2/(rho*g)));
lambda:=subs(Param,subs(delta=TTZ,1-delta/(delta-1)));
A:=7; Epsilon:=A*kappa;
omega0:=evalf(subs(Param, sqrt(lambda*g*kappa)));
xiT_2:=evala(subs(Param,subs(BSet2,delta=TTZ,
      beta=0,alpha=a*kappa,Xi_Ser)));
etaT_2:=evala(subs(Param, subs(BSet2,delta=TTZ,
      beta=0, alpha=a*kappa,Eta_Ser)));
OM_2:=subs(delta=TTZ, epsilon=Epsilon,
            omega0 + Omega1*epsilon+Omega2*epsilon^2);
Y_2:=unapply(evalf(subs(epsilon=Epsilon,psi=OM_2*t,
            epsilon*etaT_2/kappa)),[a,t]);
X_2:=unapply(evalf(subs(epsilon=Epsilon,psi=OM_2*t,
            a+epsilon*xiT_2/kappa)),[a,t]);
animate([X_2(a,t),Y_2(a,t),a =0..50],t=0..5,
        color=blue,thickness=4,scaling=constrained,frames=300);
```

## 10.4 Integral Equations

*Integral equations* arise in various areas of science and numerous applications, for example, mathematical physics, fluid mechanics, theory of elasticity, biomechanics, economics, medicine, control theory, etc.

In *Maple* and *Mathematica*, there are no built-in functions to find exact, approximate analytical and numerical solutions of integral equations. Following the analytical approach, [19], [25], [26], we show

how to solve the most important integral equations with *Maple* and *Mathematica*.

*Integral equations* can be divided into two main classes: linear and non-linear integral equations. In general, linear integral equations can be written as follows:

$$\beta f(x) + \int_D K(x, t)f(t) dt = g(x), \quad x \in D, \quad (10.6)$$

where  $f(x)$  is the unknown function;  $\beta$  (the coefficient),  $K(x, t)$  (the kernel),  $g(x)$  (the free term or the right-hand side of the integral equation) are given functions of the integral equation;  $D$  is a bounded or unbounded domain in a finite-dimensional Euclidean space,  $x$  and  $t$  are points of this space, and  $dt$  is the volume element. It is required to determine  $f(x)$  such that Eq.(10.6) holds for all (or almost all, if the integral is taken in the sense of Lebesgue)  $x \in D$ . If  $g(x) \equiv 0$ , then the integral equation is said to be homogeneous, otherwise it is called inhomogeneous.

There are three distinct types of linear integral equations, depending on the coefficient  $\beta$ :

- (i)  $\beta = 0$  for all  $x \in D$ , then Eq. (10.6) is called an equation of the first kind;
- (ii)  $\beta \neq 0$  for all  $x \in D$ , an equation of the second kind;
- (iii)  $\beta = 0$  on some non-empty subset of  $S \subset D$ , an equation of the third kind.

We consider the most important linear integral equations of the first and second kind in the one-dimensional case with variable integration limit and constant limits of integration, respectively:

$$\beta f(x) - \lambda \int_a^x K(x, t)f(t) dt = g(x), \quad K(x, t) \equiv 0, \quad t > x, \quad (10.7)$$

$$\beta f(x) - \lambda \int_a^b K(x, t)f(t) dt = g(x), \quad (10.8)$$

where  $x \in [a, b]$ ,  $\lambda$  is the parameter of integral equations.

If the kernels and the right-hand sides of integral equations satisfy special conditions, then the integral equations (10.7) and (10.8) are called, respectively, the Volterra and Fredholm equations of the first/second kind. Usually, these special conditions are: the kernel  $K(x, t)$  is continuous or square-integrable in  $\Omega = \{a \leq x \leq b, a \leq t \leq b\}$ , and  $g(x)$  is continuous or square-integrable on  $[a, b]$ .

If a linear integral equation is not of the form (10.8), then it is called a singular equation. In these equations one or both of the limits of integration are infinity or the integral is to be understood as a Cauchy principal value. The equations of the second kind arise more frequently in problems of mathematical physics.

The most important nonlinear integral equations can be written in a general form as follows:

$$\beta f(x) - \lambda \int_a^x K(x, t, f(t)) dt = g(x), \quad K(x, t, f(t)) \equiv 0, \quad t > x, \quad (10.9)$$

$$\beta f(x) - \lambda \int_a^b K(x, t, f(t)) dt = g(x), \quad (10.10)$$

where  $x \in [a, b]$ .

*Linear integral equations* of the first kind with variable integration limit:

$$\int_a^x K(x, t) f(t) dt = g(x).$$

**Problem:** Reducing the Volterra integral equation of the first kind with  $g(x) = x$ ,  $K(x, t) = \exp(x)$ , to the Volterra integral equation of the second kind and find the exact solution of the integral equation. Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
with(inttrans): g:=x->x; K:=(x,t)->exp(x); a:=0;
Eq1:=Int(K(x,t)*f(t),t=a..x)=g(x);
Eq2:=expand(isolate(diff(Eq1,x),f(x)));
laplace(g(x),x,p); laplace(K(x,t),x,p); laplace(f(x),x,p);
Eq2_L:=laplace(Eq2,x,p);
Sol:=factor(solve(subs(laplace(f(x),x,p)=F(p),Eq2_L),F(p)));
f:=x->invlaplace(Sol,p,x); f(x);
simplify(value(rhs(Eq2)-lhs(Eq2)));
simplify(value(rhs(Eq1)-lhs(Eq1)));
```

*Mathematica:*

```

g[x_]:=x; K[x_,t_]:=Exp[x]; a=0;
h[u_*v_]:=u*Integrate[v,{t,a,x}]/;FreeQ[u,v]
Eq1=h[K[x,t]*f[t]]==g[x]
Eq2=Solve[D[Eq1,x],f[x]]//FullSimplify
{LaplaceTransform[g[x],x,p],LaplaceTransform[K[x,t],x,p],
 LaplaceTransform[f[x],x,p]}
{Eq2Lap=LaplaceTransform[Eq2,x,p]/
 {LaplaceTransform[f[x],x,p]->F[p]},
 Eq3=Eq2Lap[[1,1,1]]==Eq2Lap[[1,1,2]],
 Sol=Solve[Eq3,F[p]],
 InvLap=Map[InverseLaplaceTransform[#,p,x]&,Sol,{3}]}
f[t_]:=InvLap[[1,1,2]]/.{x->t}; Eq1

```

**Problem:** Applying the Laplace transform, solve the Volterra integral

equation of the first kind of convolution type  $\int_0^x K(x-t)f(t) dt = g(x)$ ,

$g(x) = x^n$ ,  $K(x-t) = \exp(x-t)$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```

with(inttrans): n:=2; g:=x->x^n; K:=(x,t)->exp(x-t); a:=0;
Eq1:=Int(K(x,t)*f(t),t=a..x)=g(x);
laplace(g(x),x,p); laplace(K(x,t),x,p); laplace(f(x),x,p);
Eq1_L:=laplace(Eq1,x,p);
Sol:=factor(solve(subs(laplace(f(x),x,p)=F(p),Eq1_L),F(p)));
f:=x->invlaplace(Sol,p,x); f(x); Eq1;
simplify(value(Eq1));

```

*Mathematica:*

```

n=2; g[x_]:=x^n; K[x_,t_]:=Exp[x-t]; a=0
Eq1=Integrate[K[x,t]*f[t],{t,a,x}]==g[x]
{LaplaceTransform[g[x],x,p], LaplaceTransform[K[x,t],x,p],
 LaplaceTransform[f[x],x,p]}
Eq1Lap=LaplaceTransform[Eq1,x,p]/
 {LaplaceTransform[f[x],x,p]->F[p]}
Sol=Solve[Eq1Lap,F[p]]
InvLap=Map[InverseLaplaceTransform[#,p,x]&,Sol,{3}]
f[t_]:=InvLap[[1,1,2]]/.{x->t}; Eq1

```

*Linear integral equations* of the second kind with variable integration limit:

$$f(x) - \lambda \int_a^x K(x, t)f(t) dt = g(x).$$

**Problem:** Show that the function  $f(x) = xe^x$  is the solution of the Volterra integral equation of the second kind with  $g(x)=\sin(x)$ ,  $K(x, t)=\cos(x - t)$ ,  $\lambda = 2$ .

*Maple:*

```
lambda:=2: a:=0; f:=x->x*exp(x); g:=x->sin(x);
K:=(x,t)->cos(x-t);
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..x)=g(x);
value(Eq1);
```

*Mathematica:*

```
f[x_]:=x*Exp[x]; g[x_]:=Sin[x]; K[x_,t_]:=Cos[x-t];
{lambda=2, a=0,
 Eq1=f[x]-lambda*Integrate[K[x,t]*f[t],{t,a,x}]==g[x]}
```

**Problem:** Construct the Volterra integral equation of the second kind corresponding to the linear ordinary differential equation with constant coefficients  $y''_{xx} + xy'_x + y = 0$ .

*Maple:*

```
a:=0; DifEq:=diff(y(x),x$2)+x*diff(y(x),x)+y=0;
IC:=[1,0]; Eq1:=diff(y(x),x$2)=f(x);
Eq11:=subs({_C1=IC[2]}, diff(y(x),x)=int(f(t),t=a..x)+_C1);
Eq2:=dsolve(Eq1, y(x)); EqSub:=Int(Int(f(x),x),x)=
    subs(n=2, 1/(n-1)!*int((x-t)^(n-1)*f(t),t=a..x));
Eq3:=subs(op(1,rhs(Eq2))=rhs(EqSub), Eq2);
Eq4:=y=rhs(subs({_C1=IC[2], _C2=IC[1]}, Eq3));
Eq5:=subs({Eq1, Eq11, Eq4}, DifEq);
IntEq:=factor(combine(isolate(Eq5, f(x))));
```

*Mathematica:*

```
{a=0, DifEq=y''[x]+x*y'[x]+y[x]==0, IC={1,0}}
fun[x_]:=f[x]; Eq1=y''[x]->fun[x]
Eq11=y''[x]->Integrate[fun[t],{t,a,x}]+C[2]/.{C[2]->IC[[2]]}
Eq2=DSolve[{y''[x]==fun[x]},y[x],x];
Eq21=Eq2/.{Eq2[[1,1,2,3]]->1/(n-1)!*Integrate[(x-t)^(n-1)
    *fun[t],{t,a,x}]}/{n->2}]
{Eq3=Eq21[[1,1]]/.{C[1]->IC[[1]],C[2]->IC[[2]]},
 Eq4=DifEq/.{Eq1,Eq11,Eq3}, IntEq=Solve[Eq4,f[x]],
 IntEq[[1,1,1]]==IntEq[[1,1,2]]}
```

**Problem:** Applying the resolvent kernel method to the Volterra integral equation of the second kind with  $\lambda = 1$ ,  $a = 0$ ,  $g(x) = \exp(x^2)$ ,  $K(x, t) = \exp(x^2 - t^2)$ , construct the resolvent kernel  $R(x, t)$  and the exact solution. Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
K:=(x,t)->exp(x^2-t^2); K1:=(x,t)->K(x,t); g:=x->exp(x^2);
lambda:=1; a:=0; k:=10; for i from 2 to k do
    K||i:=unapply(factor(value(int(K(x,s)*K||(i-1)(s,t),
        s=t..x))), x, t); print(K||i(x,t)); od:
R:=unapply(sum('exp(x^2-t^2)*(x-t)^n/n!', 
    'n'=0..infinity), x,t);
Sol:=unapply(g(x)+lambda*int(R(x,t)*g(t),t=a..x),x);
Eq1:=value(Sol(x)-lambda*Int(K(x,t)*Sol(t),t=a..x)=g(x));
```

*Mathematica:*

```
K[x_,t_]:=Exp[x^2-t^2]; KP[x_,t_]:=K[x,t]; g[x_]:=Exp[x^2];
{lambda=1, a=0, k=10, L=Array[x,k,0], M=Array[x,k,0]}
{L[[1]]=KP[x,t], M[[1]]=KP[s,t]}
Do[ KP[X_,T_]:=Integrate[K[x,s]*M[[i-1]],
    {s,t,x}]/.{x->X,t->T}; L[[i]]=KP[x,t];
    M[[i]]=KP[s,t], {i,2,k}];
T1=Table[L[[i]],[i,1,k]]//FullSimplify
R[X_,T_]:=Sum[Exp[x^2-t^2]*(x-t)^n/n!,
    {n,0,Infinity}]/.{x->X,t->T};
Sol[X_]:=g[x]+lambda*Integrate[
    R[x,t]*g[t],{t,a,x}]/.{x->X}; Factor[Sol[x]]
Eq1=Sol[x]-lambda*Integrate[K[x,t]*Sol[t],{t,a,x}]==g[x]
```

**Problem:** Representing the kernel  $K(x, t) = (x - t)$  of the Volterra integral equation of the second kind (with  $\lambda = 1$ ,  $g(x) = \sin(x)$ ) as a polynomial of the order  $n - 1$  and solving the corresponding initial value problem for ODE, find the resolvent kernel  $R(x, t)$  and construct the exact solution.

*Maple:*

```

n:=10; K:=(x,t)->x-t; L:=NULL; lambda:=1; a:=0;
g:=x->sin(x); K_P:=a0+add(a||i*(x-t)^i,i=1..n-1);
for i from 0 to n-1 do
  if i=1 then a||i:= 1 else a||i :=0 fi:
  L := L,a||i: od: L:=[L];
H := unapply(convert(rhs(simplify(dsolve(
  {diff(h(x), x$2)-h(x)=0,
   h(t)=0, D(h)(t)=1},h(x)))),trig), x);
R := unapply(1/lambda*diff(H(x),x$2),x,t);
Sol:=unapply(g(x)+lambda*int(R(x,t)*g(t),t=a..x),x);
Eq1:=value(Sol(x)-lambda*Int(K(x,t)*Sol(t),t=a..x)=g(x));

```

*Mathematica:*

```

K[x_,t_]:=x-t; g[x_]:=Sin[x]; x[i_]:=0;
{n=10,L=Array[x,n,0],ck=Array[x,n,0],lambda=1,a=0}
KP=A[0]+Sum[A[i]*(x-t)^i,{i,1,n-1}]==K[x,t]
Do[ck[[i]]=Coefficient[KP[[1]],A[i-1]]/KP[[2]];
  If[ck[[i]]\!\Element Reals,L[[i]]=ck[[i]],L[[i]]=0],
  {i,1,n}];
T1=Table[L[[i]],[i,1,n]]
solDif=DSolve[{h''[x]-h[x]==0,h[t]==0,h'[t]==1},h[x],x]
H[X_]:=FullSimplify[ExpToTrig[solDif[[1,1,2]]]]/.{x->X};
R[X_,T_]:=1/lambda*H''[x]/.{x->X,t ->T}; {H[X], R[x,t]}
Sol[X_]:=g[x]+lambda*Integrate[R[x,t]*g[t],
  {t,a,x}]/.{x->X}; Sol[x]
Eq1=Sol[x]-lambda*Integrate[K[x,t]*Sol[t],
  {t,a,x}]==g[x]/.FullSimplify

```

**Problem:** Applying the Laplace transform to the Volterra integral equation of the second kind of convolution type,  $K(x - t) = e^{-(x-t)} \sin(x - t)$ ,  $\lambda = 1$ ,  $g(x) = \cos x$ , and using the Convolution Theorem for the Laplace transform, find the resolvent kernel of this equation and construct the exact solution. Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
with(inttrans): lambda :=1; g:=x->cos(x);
K:=(x,t)->exp(-(x-t))*sin(x-t); a:=0;
K_L:=laplace(exp(-X)*sin(X),X,p);
R_L:=factor(K_L/(1-K_L));
R:=unapply(invlaplace(R_L,p,X), X); R(x-t);
Sol:=unapply(g(x)+lambda*int(R(x-t)*g(t),t=a..x),x);
Eq1:=combine(value(Sol(x)-lambda*Int(K(x,t)*Sol(t),
t=a..x))=g(x));
```

*Mathematica:*

```
lambda=1; g[x_]:=Cos[x]; K[x_,t_]:=Exp[-(x-t)]*Sin[x-t];
{a=0, KL=LaplaceTransform[Exp[-X]*Sin[X],X,p],
 RL=Factor[KL/(1-KL)]}
R[z_]:=InverseLaplaceTransform[RL,p,X]/.{X->z}; R[x-t]
Sol[z_]:=g[x]+lambda*
    Integrate[R[x-t]*g[t],{t,a,x}]/.{x->z}; Sol[x]
Eq1=FullSimplify[Sol[x]-lambda*Integrate[K[x,t]*Sol[t],
 {t,a,x}]==g[x]]
```

**Problem:** Applying the Laplace transform find the exact solution of the Volterra integral equation of the second kind of convolution type,  $g(x) = \sinh(x)$ ,  $K(x - t) = \cosh(x - t)$ ,  $\lambda = 1$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
with(inttrans): lambda:=1; a:=0; g:=x->sinh(x);
K:=(x,t)->cosh(x-t); laplace(g(x),x,p);
laplace(K(x,t),x,p); laplace(f(x),x,p);
Eq1:=f(x)-lambda*int(K(x,t)*f(t),t=a..x)=g(x);
Eq1_L:=laplace(Eq1,x,p);
Sol:=factor(solve(subs(
    laplace(f(x),x,p)=F(p),Eq1_L), F(p)));
f:=x->invlaplace(Sol,p,x); f(x);
B1:=convert(Eq1,exp); simplify(rhs(B1)-lhs(B1));
```

*Mathematica:*

```

g[x_]:=Sinh[x]; K[x_,t_]:=Cosh[x-t]; {lambda=1,a=0}
{LaplaceTransform[g[x],x,p], LaplaceTransform[K[x,t],x,p],
 LaplaceTransform[f[x],x,p]}
Eq1=f[x]-lambda*Integrate[K[x,t]*f[t],{t,a,x}]==g[x]
Eq1Lap=LaplaceTransform[Eq1,x,p].
    {LaplaceTransform[f[x],x,p]→F[p]}
Sol:=Solve[Eq1Lap,F[p]]
InvLap=Map[InverseLaplaceTransform[#,p,x]&,Sol,{3}]
f[t_]:=FullSimplify[ExpToTrig[InvLap[[1,1,2]]/.{x→t}]];
{f[x], Eq1}

```

*Linear integral equations* of the first kind with constant limits of integration

$$\int_a^b K(x,t) f(t) dt = g(x).$$

We note that if  $K(x,t)$  is a square integrable function in  $\Omega$ ,  $g(x) \in L^2(a,b)$ ,  $f(x) \in L^2(a,b)$ , the problem of finding solutions of linear integral equations of the first kind with constant limits of integration belongs to the class of ill-posed problems, i.e. this problem is unstable with respect to small changes of the right-hand side of the integral equation.

*The most important methods* for studying linear integral equations of the first kind are the methods for constructing approximate solutions of ill-posed problems. Here we consider the successive approximations method and the regularization method.

**Problem:** Applying the successive approximations method solve the Fredholm integral equation of the first kind with  $f_0(x) = 0$ ,  $a = 0$ ,  $b = 1$ ,  $g(x) = 1$ ,  $K(x,t) = 1$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```

k:=10; f0:=x->0; g:=x->1; a:=0; b:=1;
lambda:=1; K:=(x,t)->1; for i from 1 to k do
f||i:=unapply(f||(i-1)(x)+lambda*(g(x)
    -int(K(x,t)*f||(i-1)(t),t=a..b)),x); od:
for i from 0 to k-1 do simplify(f||i(x)); od;
Sol:=x->1; Eq1:=value(lambda*Int(K(x,t)*Sol(t),t=a..b)=g(x));

```

*Mathematica:*

```
fP[x_]:=0; g[x_]:=1; K[x_,t_]:=1;
{k=10, a=0, b=1, lambda=1, L=Array[x,k,0],
 M=Array[x,k,0], L[[1]]=fP[x], M[[1]]=fP[t]}
Do[fP[X_]:=L[[i-1]]+lambda*(g[x]-Integrate[K[x,t]*M[[i-1]],
 {t,a,b}])/.{x->X};
 L[[i]]=fP[x]; M[[i]]=fP[t], {i,2,k}];
T1=Table[L[[i]],{i,1,k}]//FullSimplify
Sol[x_]:=1; Eq1=lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]==g[x]
```

**Problem:** Applying the Tikhonov regularization method solve the Fredholm integral equation of the first kind with  $a = 0$ ,  $b = 1$ ,  $g(x) = 1$ ,  $K(x, t) = 1$ ,  $\lambda = 1$ .

*Maple:*

```
K:=(x,t)->1; K1:=(x,t)->K(x,t); g:=x->1; lambda:=1; a:=0; b:=1;
for i from 2 to 10 do
K||i:=unapply(factor(value(int(K(x,s)*K||(i-1)(s,t),
s=a..b))),x,t); print(K||i(x,t)); od:
R:=unapply(sum('1*(lambda/epsilon)^(n-1)',n'=1..infinity),x,t);
Sol:=unapply(g(x)/epsilon+lambda/epsilon
*int(R(x,t)*g(t),t=a..b),x);
Eq1:=factor(value(epsilon*Sol(x)-lambda*Int(K(x,t)
*Sol(t),t=a..b)=g(x)));
Eq2:=factor(value(lambda*Int(K(x,t)*Sol(t),t=a..b)));
B:=sqrt(int((g(x)-Eq2)^2,x=a..b)); subs(epsilon=0.382,B);
plot(B,epsilon=0.38..0.4); subs(epsilon=0.382, Sol(x));
```

*Mathematica:*

```
K[x_,t_]:=1; KP[x_,t_]:=K[x,t]; g[x_]:=1;
{lambda=1, a=0, b=1, k=10, L=Array[x,k,0], M=Array[x,k,0]}
{L[[1]]=KP[x,t], M[[1]]=KP[s,t]}
Do[KP[X_,T_]:=Integrate[K[x,s]*M[[i-1]],[s,a,b]]/.{x->X,t->T};
L[[i]]=KP[x,t]; M[[i]]=KP[s,t],{i,2,k}];
T1=Table[L[[i]],{i,1,k}]//FullSimplify
R[X_,T_]:=Sum[1*(lambda/[\[Epsilon]])^(n-1),
{n,1,Infinity}]/.{x->X,t->T};
Sol[X_]:=g[x]/[\[Epsilon]]+lambda/[\[Epsilon]]*Integrate[
R[x,t]*g[t],{t,a,b}]/.{x->X}; Factor[Sol[x]]
```

```

{Eq1=\[Epsilon]*Sol[x]-lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]
 ==g[x], Eq2=lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]}
B=Sqrt[Integrate[(g[x]-Eq2)^2,{x,a,b}]]
B/.{\[Epsilon]\[Rightarrow]0.382}
Plot[B,{\[Epsilon],0.38,0.4},PlotRange\[Rightarrow]All,PlotStyle\[Rightarrow]Red,
 Frame\[Rightarrow]True]; Sol[x]/.\[Epsilon]\[Rightarrow]0.382]

```

*Linear integral equations* of the second kind with constant limits of integration

$$f(x) - \lambda \int_a^b K(x, t) f(t) dt = g(x).$$

**Problem:** Show that the function  $f(x) = \cos(2x)$  is the solution of the Fredholm integral equation of the second kind with

$$g(x) = \cos(x), \quad \lambda = 3, \quad a = 0, \quad b = \pi, \quad K(x, t) = \begin{cases} \sin x \cos t & a \leq x \leq t, \\ \sin t \cos x & t \leq x \leq b. \end{cases}$$

*Maple:*

```

a:=0; b:=Pi; f:=x->cos(2*x); g:=x->cos(x); lambda:=3;
K:=(x,t)->piecewise(x>=a and x <=t, sin(x)*cos(t),
                      x>t and x<=b, sin(t)*cos(x));
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..b,'AllSolutions')=g(x);
B1:=combine(value(Eq1)) assuming x>a and x<b;

```

*Mathematica:*

```

f[x_]:=Cos[2*x]; g[x_]:=Cos[x]; {a=0,b=Pi,lambda=3}
K[x_,t_]:=Piecewise[{{Sin[x]*Cos[t],a <=x<=t},
                      {Sin[t]*Cos[x],t<x<=b}}];
Eq1=f[x]-lambda*Integrate[K[x,t]*f[t],{t,a,b}]==g[x]
B1=FullSimplify[Assuming[x>a && x<b,PiecewiseExpand[Eq1]]]

```

**Problem:** Applying the Fredholm determinant method, construct the Fredholm resolvent kernel  $R(x, y; \lambda)$  of the Fredholm integral equation of the second kind with  $a=0$ ,  $b=1$ ,  $g(x)=\exp(-x)$ ,  $K(x, t)=x \exp(t)$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```

with(LinearAlgebra): a:=0; b:=1; g:=x->exp(-x);
K:=(x,t)->x*exp(t); lambda:=2;
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..b)=g(x);
B0:=K(x,t); C0:=1;
for k from 2 to 5 do
  DF||k :=Matrix(1..k,1..k, []);
  DF||k[1,1]:=K(x,t);
  i:=1; for j from 2 to k do DF||k[i,j]:=K(x,t||(j-1)) od;
  j:=1; for i from 2 to k do DF||k[i,j]:=K(t||(i-1),t) od;
  for i from 2 to k do for j from 2 to k do
    DF||k[i,j]:=K(t||(i-1),t||(j-1)) od; od:
od: DF2; DF3; DF4; DF5;
B1:= value(Int(Determinant(DF2),t1=a..b));
B2:= value(Int(Int(Determinant(DF3),t1=a..b),t2=a..b));
B3:= value(Int(Int(Int(Determinant(DF4),
  t1=a..b),t2=a..b),t3=a..b));
B4:= value(Int(Int(Int(Int(Determinant(DF5),
  t1=a..b),t2=a..b),t3=a..b),t4=a..b));
for k from 1 to 4 do
  DDF||k :=Matrix(1..k,1..k, []);
  for i from 1 to k do for j from 1 to k do
    DDF||k[i,j]:=K(t||(i),t||(j)) od; od:
od:
DDF1; DDF2; DDF3; DDF4;
C1:= value(Int(Determinant(DDF1),t1=a..b));
C2:= value(Int(Int(Determinant(DDF2),t1=a..b),t2=a..b));
C3:= value(Int(Int(Int(Determinant(DDF3),t1=a..b),
  t2=a..b),t3=a..b));
C4:= value(Int(Int(Int(Int(Determinant(DDF4),
  t1=a..b),t2=a..b),t3=a..b),t4=a..b));
DN :=K(x,t)+add((-1)^n/n!*B||n*lambda^n, n=1..3);
DD :=1+add((-1)^n/n!*C||n*lambda^n, n=1..3);
R :=unapply(DN/DD,x,t);
Sol:=unapply(value(g(x)+lambda*Int(R(x,t)*g(t),t=a..b)),x);
Eq1:=value(Sol(x)-lambda*Int(K(x,t)*Sol(t),t=a..b))=g(x));

```

*Mathematica:*

```

<<LinearAlgebra`MatrixManipulation`;
g[x_]:=Exp[-x]; K[x_,t_]:=x*Exp[t];
{a=0,b=1,n=5,lambda=2}

```

```

{BA=Array[x1,n,0],CA=Array[x2,n,0],tt=Array[t,n,1],
 BA[[0]]=K[x,t], CA[[0]]=1}
Eq1=f[x]-lambda*Integrate[K[x,t]*f[t],{t,a,b}]==g[x]
DFF[z_]:=Array[x4,{z,z},{1,1}];
DA[k_]:=Module[{},

DF=Array[x5,{k,k},{1,1}];DF[[1,1]]=K[x,t];
i=1; Do[DF[[i,j]]=K[x,tt[[j-1]]],{j,2,k}];
j=1; Do[DF[[i,j]]=K[tt[[i-1]],t],{i,2,k}];
Do[Do[DF[[i,j]]=K[tt[[i-1]],tt[[j-1]]],
{i,2,k}],{j,2,k}]; DF];

Map[MatrixForm,{DA[2],DA[3],DA[4],DA[5]}]
BA[[1]]=Integrate[Det[DA[2]],{t[1],a,b}]
BA[[2]]=Integrate[Integrate[Det[DA[3]],{t[1],a,b}],{t[2],a,b}]
BA[[3]]=Integrate[Integrate[Integrate[Det[DA[4]],
{t[1],a,b}],{t[2],a,b}],{t[3],a,b}]
BA[[4]]=Integrate[Integrate[Integrate[Integrate[Det[DA[5]],
{t[1],a,b}],{t[2],a,b}],{t[3],a,b}],{t[4],a,b}]
DDA[k_]:=Module[{},

DDF=Array[x6,{k,k},{1,1}];
Do[Do[DDF[[i,j]]=K[tt[[i]],tt[[j]]],
{i,1,k}],{j,1,k}]; DDF];

Map[MatrixForm,{DDA[1],DDA[2],DDA[3],DDA[3]}]
CA[[1]]=Integrate[Det[DDA[1]],{t[1],a,b}]
CA[[2]]=Integrate[Integrate[Det[DDA[2]],
{t[1],a,b}],{t[2],a,b}]
CA[[3]]=Integrate[Integrate[Integrate[Det[DDA[3]],
{t[1],a,b}],{t[2],a,b}],{t[3],a,b}]
CA[[4]]=Integrate[Integrate[Integrate[Integrate[
Det[DDA[4]],{t[1],a,b}],{t[2],a,b}],
{t[3],a,b}],{t[4],a,b}]

DN=BA[[0]]+Sum[(-1)^n/n!*BA[[n]]*lambda^n,{n,1,3}]
DD=CA[[0]]+Sum[(-1)^n/n!*CA[[n]]*lambda^n,{n,1,3}]
R[X_,T_]:=(DN/DD).{x->X,t->T}; R[x,t]
Sol[X_]:=g[x]+lambda*Integrate[R[x,t]*g[t],
{t,a,b}]/.{x->X}; Sol[x]
Eq1=Sol[x]-lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]==g[x]

```

**Problem:** Applying the Fredholm determinant method and the recurrence relations, construct the Fredholm resolvent kernel  $R(x, y; \lambda)$  of the Fredholm integral equation of the second kind with  $a = 0$ ,  $b = 1$ ,  $g(x) = x$ ,  $K(x, t) = x - 2t$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
a:=0; b:=1; g:=x->x; K:=(x,t)->x-2*t; lambda:=1;
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..b)=g(x);
B0:=(x,t)->K(x,t); C0:=1;
for k from 1 to 9 do
C||k := value(Int(B||(k-1)(s,s),s=a..b));
B||k := unapply(value(C||k*K(x,t)
-k*Int(K(x,s)*B||(k-1)(s,t),s=a..b)),x,t); od;
DN :=K(x,t)+add((-1)^n/n!*B||n(x,t)*lambda^n, n=1..9);
DD :=1+add((-1)^n/n!*C||n*lambda^n, n=1..9);
R :=unapply(DN/DD,x,t);
Sol:=unapply(value(g(x)+lambda*Int(R(x,t)*g(t),t=a..b)),x);
Eq1:=value(Sol(x)-lambda*Int(K(x,t)*Sol(t),t=a..b))=g(x);
```

*Mathematica:*

```
g[x_]:=x; K[x_,t_]:=x-2*t; KP[x_,t_]:=K[x, t];
{a=0,b=1,lambda=1,n=9,BL=Array[x1,n,0],
 BM=Array[x2,n,0],BN=Array[x3,n,0],CA=Array[x4,n,0],
 BL[[0]]=KP[s,t],BM[[0]]=KP[s,s],BN[[0]]=KP[x,t],
 CA[[0]]=1}
Eq1=f[x]-lambda*Integrate[K[x,t]*f[t],{t,a,b}]==g[x]
Do[CA[[k]]=Integrate[BM[[k-1]],{s,a,b}],
 KP[X_,T_]:=CA[[k]]*K[x,t]
-k*Integrate[K[x,s]*BL[[k-1]],[s,a,b]}/.{x->X,t->T};
 BL[[k]]=KP[s,t]; BM[[k]]=KP[s,s]; BN[[k]]=KP[x,t],
 {k,1,n}];
T1=Table[BN[[i]],{i,1,n}]//FullSimplify
T2=Table[CA[[i]],{i,1,n}]//FullSimplify
DN=BN[[0]]+Sum[(-1)^n/n!*BN[[n]]*lambda^n,{n,1,9}]
DD=CA[[0]]+Sum[(-1)^n/n!*CA[[n]]*lambda^n,{n,1,9}]
R[X_,T_]:=(DN/DD)/.{x->X,t->T}; R[x,t]
Sol[X_]:=g[x]+lambda*Integrate[R[x,t]*g[t],
 {t,a,b}]/.{x->X}; Sol[x]
Eq1=Sol[x]-lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]==g[x]
```

**Problem:** Applying the resolvent kernel method to the Fredholm integral equation of the second kind with  $g(x) = x$ ,  $K(x, t) = xt$ ,  $a = 0$ ,  $b = 1$ , construct the resolvent kernel  $R(x, t; \lambda)$  and the exact solution. Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
K:=(x,t)->x*t; K1:=(x,t)->K(x,t); g:=x->x; a:=0; b:=1;
for i from 2 to 10 do
  K||i:=unapply(factor(value(int(K(x,s)*K||(i-1)(s,t),s=a..b))),x,t): print(K||i(x,t));
od;
R:=unapply(sum(`(x*t)/3^(n-1)*lambda^(n-1)`,'n'=1..infinity),x,t,lambda);
Sol:=unapply(g(x)+lambda*int(R(x,t,lambda)*g(t),t=a..b),x);
Eq1:=simplify(value(Sol(x)-lambda*Int(K(x,t)*Sol(t),t=a..b)=g(x)));
B:=sqrt(int(int(K(x,t)^2,x=a..b),t=a..b)); abs(lambda)<1/B;
```

*Mathematica:*

```
K[x_,t_]:=x*t; KP[x_,t_]:=K[x,t]; g[x_]:=x;
{a=0,b=1,n=10,L=Array[x1,n,1],M=Array[x2,n,1],
 L[[1]]=KP[s,t],M[[1]]=KP[x,t]}
Do[KP[X_,T_]:=Integrate[K[x,s]*L[[i-1]],[s,a,b]]/.{x->X,t->T},
 L[[i]]=KP[s,t]; M[[i]]=KP[x,t], {i,2,n}];
T1=Table[M[[i]],{i,1,n}]//FullSimplify
R[X_,T_,Lambda_]:=Sum[(x*t)/3^(n-1)*lambda^(n-1),
 {n,1,Infinity}]/.{x->X,t->T,lambda->Lambda};
Sol[X_]:=g[x]+lambda*Integrate[R[x,t,lambda]*g[t],{t,a,b}]/.{x->X}; {R[x,t,lambda], Sol[x]}
{Eq1=Sol[x]-lambda*Integrate[K[x,t]*Sol[t],{t,a,b}]==g[x],
 B=Sqrt[Integrate[Integrate[K[x,t]^2,{x,a,b}],{t,a,b}]],
 Abs[lambda]<1/B}
```

**Problem:** Let  $a = -1$ ,  $b = 1$ . Verify that the kernels  $KA(x, t) = xt$  and  $KB(x, t) = x^2t^2$  are orthogonal kernels in  $[a, b]$ . Applying the resolvent kernel method to the Fredholm integral equation of the second kind with  $g(x) = x$ ,  $K(x, t) = KA(x, t) + KB(x, t)$ , construct the resolvent kernel  $R(x, t; \lambda)$  and the exact solution. Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
KA:=(x,t)->x*t; KA1:=(x,t)->KA(x,t); KB:=(x,t)->x^2*t^2;
KB1:=(x,t)->KB(x,t); g:=x->x; a:=-1; b:=1;
int(KA(x,s)*KB(s,t),s=a..b); int(KB(x,s)*KA(s,t),s=a..b);
```

```

for i from 2 to 10 do
  KA||i:=unapply(factor(value(int(KA(x,s)*KA||(i-1)(s,t),
  s=a..b))), x, t): print(KA||i(x,t));
  KB||i:=unapply(factor(value(int(KB(x,s)*KB||(i-1)(s,t),
  s=a..b))), x, t): print(KB||i(x,t)); od:
RA:=unapply(sum('(x*t)*2^(n-1)/3^(n-1)*lambda^(n-1)',
  'n'=1..infinity),x,t,lambda);
RB:=unapply(sum('(x^2*t^2)*2^(n-1)/5^(n-1)*lambda^(n-1)',
  'n'=1..infinity),x,t,lambda);
Sol:=unapply(g(x)+lambda*int(
  (RA(x,t,lambda)+RB(x,t,lambda))*g(t),t=a..b),x);
Eq1:=simplify(value(Sol(x)-lambda*Int((KA(x,t)+KB(x,t))
  *Sol(t),t=a..b)=g(x)));
B:=sqrt(int(int((KA(x,t))^2,x=a..b),t=a..b));abs(lambda)<1/B;

```

*Mathematica:*

```

KA[x_,t_]:=x*t; KPA[x_,t_]:=KA[x,t];
KB[x_,t_]:=x^2*t^2; KPB[x_,t_]:=KB[x,t]; g[x_]:=x;
{a=-1,b=1,n=10,LA=Array[x1,n,1],MA=Array[x2,n,1],
LB=Array[x3,n,1],MB=Array[x4,n,1],LA[[1]]=KPA[s,t],
MA[[1]]=KPA[x,t],LB[[1]]=KPB[s,t],MB[[1]]=KPB[x,t]}
Integrate[KA[x,s]*KB[s,t],{s,a,b}]
Integrate[KB[x,s]*KA[s,t],{s,a,b}]
Do[KPA[X_,T_]:=Integrate[KA[x,s]*LA[[i-1]],[s,a,b]]/.{x->X,t->T},
  LA[[i]]=KPA[s,t]; MA[[i]]=KPA[x,t];
  KPB[X_,T_]:=Integrate[KB[x,s]*LB[[i-1]],[s,a,b]]/.{x->X,t->T};
  LB[[i]]=KPB[s,t]; MB[[i]]=KPB[x,t], {i,2,n}];
T1=Table[MA[[i]],[i,1,n]]//FullSimplify
T2=Table[MB[[i]],[i,1,n]]//FullSimplify
RA[X_,T_,Lambda_]:=Sum[(x*t)*2^(n-1)/3^(n-1)*lambda^(n-1),
  {n,1,Infinity}]/.{x->X,t->T,lambda->Lambda};
RB[X_,T_,Lambda_]:=Sum[(x^2*t^2)*2^(n-1)/5^(n-1)
  *lambda^(n-1),{n,1,Infinity}]/.{x->X,t->T,lambda->Lambda};
Sol[X_]:=g[x]+lambda*Integrate[(RA[x,t,lambda]
  +RB[x,t,lambda])*g[t],{t,a,b}]/.{x->X};
{RA[x,t,lambda], RB[x,t,lambda], Sol[x]}
{Eq1=FullSimplify[Sol[x]-lambda*Integrate[(KA[x,t]+KB[x,t])
  *Sol[t],{t,a,b}]]==g[x]},
B=Square[Integrate[Integrate[KA[x,t]^2,{x,a,b}],{t,a,b}]],
Abs[lambda]<1/B}

```

**Problem:** Let  $a = -\pi$ ,  $b = \pi$ . Solve the Fredholm integral equation of the second kind with the degenerate kernel  $K(x, t) = x \cos t + t^2 \sin x + \cos x \sin t$ , and  $g(x) = x$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
with(LinearAlgebra): g:=x->x; a:=-Pi; b:=Pi;
K:=(x,t)->x*cos(t)+t^2*sin(x)+cos(x)*sin(t);
k:=nops(K(x,t)); var:=[C1,C2,C3];
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..b)=g(x);
Eq2:=f(x)-add(lambda*Int(op(i,K(x,t)))*f(t),
    t=a..b), i=1..k)=g(x); Eq3:= isolate(Eq2,f(x));
CC1:= expand(op(2,rhs(Eq3))/x/lambda);
CC2:= expand(op(3,rhs(Eq3))/sin(x)/lambda);
CC3:= expand(op(4,rhs(Eq3))/cos(x)/lambda);
F := lambda*x*C1+lambda*sin(x)*C2+lambda*cos(x)*C3+g(x);
Eq4:= subsop(2=F, Eq3); f:= unapply(rhs(Eq4),x);
sys := {}: for i from 1 to 3 do CC||i;
    sys:= sys union {value(expand(C||i=CC||i))}; od;
sys:=convert(sys,list);
M1:= GenerateMatrix(sys,var,augmented=true);
(M2,b2):= GenerateMatrix(sys,var); Determinant(M2);
b1:=LinearSolve(M1);subC := {};
for i from 1 to k do subC:=subC union {C||i=b1[i]}; od;
Sol:=unapply(subs(subC,F),x); Eq11:=simplify(value(Sol(x)
-lambda*Int(K(x,t)*Sol(t),t=a..b)=g(x)));
```

*Mathematica:*

```
f1={x,Sin[x],Cos[x]}; f2={Cos[t],t^2,Sin[t]};
g[x_]:=x; {a=-Pi,b=Pi}
h[u_*v_]/;FreeQ[u,v]:=u*Integrate[v,{t,a,b}];
K[x_,t_]:=Plus@@Times[f1,f2]; K[x,t]
{k=Length[K[x,t]], CA=Array[c,k,1], CC=Array[x1,k,1]}
Eq11=f[x]-Sum[lambda*h[K[x,t][[i]]*f[t]],{i,1,k}]==g[x]
ClearAttributes[{Times,Plus},Orderless];
Eq1=f[x]-Sum[lambda*h[K[x,t][[i]]*f[t]],{i,1,k}]==g[x]
Eq2=Solve[Eq1,f[x]]
CC[[1]]=Cancel[Eq2[[1,1,2,1]]/x/lambda]
CC[[2]]=Cancel[Eq2[[1,1,2,2]]/Sin[x]/lambda]
CC[[3]]=Cancel[Eq2[[1,1,2,3]]/Cos[x]/lambda]
```

```

F0=lambda*x*CA[[1]]+lambda*Sin[x]*CA[[2]]+lambda*Cos[x]*CA[[3]]
F=F0+g[x]
f[X_]:=F/.{x->X}; f[x]
sys={}; Do[CC[[i]], sys=Union[sys,{CA[[i]]==CC[[i]]}],{i,1,k}];
{sys, S=Solve[sys,CA]}
Sol[X_]:=Cancel[((f[x]/.S[[1]])/.{x->X}); Sol[x]
B1=Distribute[K[x,t]*Sol[t]]
Eq12=FullSimplify[
    Sol[x]-lambda*Map[Integrate[#1,{t,a,b}]&,B1]]==g[x]

```

**Problem:** Let  $a = 0$ ,  $b = \pi$ . Find the eigenvalues and eigenfunctions of the Fredholm integral equation of the second kind with the degenerate kernel  $K(x, t) = \cos^2 x \cos 2t + \cos 3x \cos^3 t$ , and  $g(x) = x$ . Construct the general solution  $f(x)$  and verify the solution constructed.

*Maple:*

```

with(LinearAlgebra): g:=x->0; a:=0; b:=Pi;
K:=(x,t)->cos(x)^2*cos(2*t)+cos(3*x)*cos(t)^3;
k:=nops(K(x,t)); var:=[C1,C2];
Eq1:=f(x)-lambda*Int(K(x,t)*f(t),t=a..b)=g(x);
Eq2:=f(x)-add(lambda*Int(op(i,K(x,t))*f(t),
    t=a..b),i=1..k)=g(x); Eq3:= isolate(Eq2,f(x));
CC1:= combine(op(1,rhs(Eq3))/cos(x)^2/lambda);
CC2:= combine(op(2,rhs(Eq3))/cos(3*x)/lambda);
F := lambda*cos(x)^2*C1+lambda*cos(3*x)*C2+g(x);
Eq4:= subsop(2=F, Eq3); f:= unapply(rhs(Eq4),x); sys:=NULL;
for i from 1 to k do CC||i;
    sys:= sys, value(combine(C||i=CC||i)); od; sys:=[sys];
(M1,b1):= GenerateMatrix(sys,var);
Eq5 := Determinant(M1)=0; EigenVal:=[solve(Eq5,lambda)];
M2:=subs(lambda=EigenVal[1],M1);
M3:=subs(lambda=EigenVal[2],M1); sys1:=M1.Vector(var)=0;
Eq6:=subs(lambda=EigenVal[1],sys1);
Eq7:=subs(lambda=EigenVal[2],sys1);
P1:={C2=0,C1=C1}; P2:={C1=0,C2=C2};
EigenFun1:= unapply(subs(P1,lambda=EigenVal[1],
    C1=1/EigenVal[1],F),x);
EigenFun2:= unapply(subs(P2,lambda=EigenVal[2],
    C2=1/EigenVal[2],F),x);
f:=(lambda,x)->piecewise(lambda=EigenVal[1],
    C*EigenFun1(x),lambda=EigenVal[2],C*EigenFun2(x),0);

```

```

combine(value(f(EigenVal[1],x)-EigenVal[1]*Int(K(x,t)*
      f(EigenVal[1],t),t=a..b)=g(x)));
combine(value(f(EigenVal[2],x)-EigenVal[2]*Int(K(x,t)*
      f(EigenVal[2],t),t=a..b)=g(x)));
value(f(Pi,x)-Pi*Int(K(x,t)*f(Pi,t),t=a..b)=g(x));

```

*Mathematica:*

```

<<LinearAlgebra`MatrixManipulation`; {a=0, b=Pi}
f1={Cos[x]^2,Cos[3*x]}; f2={Cos[2*t],Cos[t]^3}; g[x_]:=0;
h[u_*v_]/;FreeQ[u,v]:=u*Integrate[v,{t,a,b}];
K[x_,t_]:=Plus@@Times[f1,f2]; K[x,t]
{k=Length[K[x,t]],CA=Array[c,k,1],CC=Array[x1,k,1]}
Eq11=f[x]-Sum[lambda*h[K[x,t][[i]]*f[t]],{i,1,k}]==g[x]
ClearAttributes[{Times,Plus},Orderless];
Eq1=f[x]-Sum[lambda*h[K[x,t][[i]]*f[t]],{i,1,k}]==g[x]
Eq2=Solve[Eq1,f[x]]
CC[[1]]=Cancel[Eq2[[1,1,2,1]]/(Cos[x]^2)/lambda]
CC[[2]]=Cancel[Eq2[[1,1,2,2]]/Cos[3*x]/lambda]
F=lambda*(Cos[x]^2)*CA[[1]]+lambda*Cos[3*x]*CA[[2]]+g[x]
f[X_]:=F/.{x->X}; f[x]
sys0=Table[CA[[i]]==CC[[i]],{i,1,k}]
genMat=LinearEquationsToMatrices[sys0,CA]
{Mat1=genMat[[1]],b1=genMat[[2]]}
Eq5=Det[Mat1]==0
{EigenVal=Solve[Eq5, lambda], Mat2=Mat1/.EigenVal[[1]],
 Mat3=Mat1/.EigenVal[[2]], sys1=Mat1.CA==0}
{Eq6=sys1/.EigenVal[[1]], Eq7=sys1/.EigenVal[[2]]}
P1={c[2]->0,c[1]->1/EigenVal[[1,1,2]],EigenVal[[1,1]]}
P2={c[1]->0,c[2]->1/EigenVal[[2,1,2]],EigenVal[[2,1]]}
EigenFun1[X_]:=(F/.P1)/.{x->X}; EigenFun1[x]
EigenFun2[X_]:=(F/.P2)/.{x->X}; EigenFun2[x]
f[Lambda_,X_]:=Piecewise[{{
    {C*EigenFun1[x],lambda==EigenVal[[1,1,2]]},
    {C*EigenFun2[x],lambda==
        EigenVal[[2,1,2]]}}].{lambda->Lambda,x->X};
f[EigenVal[[1,1,2]],x]-EigenVal[[1,1,2]]*Integrate[K[x,t]*
      f[EigenVal[[1,1,2]],t],{t,a,b}]==g[x]
f[EigenVal[[2,1,2]],x]-EigenVal[[2,1,2]]*Integrate[K[x,t]*
      f[EigenVal[[2,1,2]],t],{t,a,b}]==g[x]
f[Pi,x]-Pi*Integrate[K[x,t]*f[Pi,t],{t,a,b}]==g[x]

```

*Nonlinear integral equations* with variable integration limit

$$\beta f(x) - \lambda \int_a^x K(x, t, f(t)) dt = g(x).$$

**Problem:** Applying the successive approximations method solve the nonlinear Volterra integral equation of the second kind with

$$f_0(x) = x, \quad \lambda = 1, \quad a = 0, \quad g(x) = 0, \quad K(x, t, f(t)) = \frac{1+f(t)^2}{1+t^2}.$$

Show that the constructed function  $f(x) = x$  is the integral equation solution.

*Maple:*

```
k:=10; lambda:=1; f0:=x->x; g:=x->0; a:=0;
for i from 1 to k do
  F||(i-1):=unapply((1+f||(i-1)(t)^2)/(1+t^2),x,t,f||(i-1));
  f||i:=unapply(g(x)+lambda*int(F||(i-1)(x,t,f||(i-1)),
    t=a..x),x); od;
f:=x->x; F:=(x,t,f)->(1+f(t)^2)/(1+t^2);
Eq1:=f(x)-lambda*Int(F(x,t,f),t=a..x)=g(x);
simplify(value(Eq1));
```

*Mathematica:*

```
fP[x_]:=x; g[x_]:=0;
{k=10,lambda=1,a=0,Lf=Array[x1,k,0],LF=Array[x2,k,0],
 Mf=Array[x3,k,0],Lf[[0]]=fP[t],Mf[[0]]=fP[x]}
Do[F[X_,T_]:=((1+Lf[[i-1]]^2)/(1+t^2))/.{x->X,t->T},
  LF[[i-1]]=F[x,t];
  ff[X_]:=g[x]+lambda*Integrate[LF[[i-1]],[t,a,x]]/.{x->X};
  Lf[[i]]=ff[t]; Mf[[i]]=ff[x],{i,1,k}];
T1=Table[Mf[[i]],{i,0,k-1}]/.FullSimplify
T2=Table[LF[[i]],{i,0,k-1}]/.FullSimplify
f[x_]:=x; F[x_,t_,f1_]:=(1+f1[t]^2)/(1+t^2); F[x,t,f1]
Eq1=f[x]-lambda*Integrate[F[x,t,f1],[t,a,x]]==g[x]
```

**Problem:** Applying the successive approximations method solve the nonlinear Volterra integral equation of the second kind with

$$f_0(x) = 0, \quad \lambda = 1, \quad a = 0, \quad g(x) = 0, \quad K(x, t, f(t)) = \frac{t^2 f(t)}{1+t^2+f(t)}.$$

Show that the  $f(x) \equiv 0$  is the integral equation solution.

*Maple:*

```

k:=10; lambda:=1; f0:=x->0; g:=x->0; a:=0;
for i from 1 to k do
  F||(i-1):=unapply((t^2*f||(i-1)(t))/(
    (1+t^2+f||(i-1)(t)),x,t,f||(i-1));
  f||i:=unapply(g(x)+lambda*
    int(F||(i-1)(x,t,f||(i-1)),t=a..x),x);
od;
f:=x->0; F:=(x,t,f)->(t^2*f(t))/(1+t^2+f(t));
Eq1:=f(x)-lambda*Int(F(x,t,f),t=a..x)=g(x);
simplify(value(Eq1));

```

*Mathematica:*

```

fP[x_]:=0; g[x_]:=0;
{k=10,lambda=1,a=0,Lf=Array[x1,k,0],LF=Array[x2,k,0],
 Mf=Array[x3,k,0],Lf[[0]]=fP[t],Mf[[0]]=fP[x]}
Do[F[X_,T_]:=((t^2*Lf[[i-1]])/(1+t^2+Lf[[i-1]]))/
  .{x->X,t->T}, LF[[i-1]]=F[x,t];
 ff[X_]:=g[x]+lambda*Integrate[LF[[i-1]},{t,a,x}]/.{x->X};
 Lf[[i]]=ff[t]; Mf[[i]]=ff[x],{i,1,k}];
T1=Table[Mf[[i]],[i,0,k-1]]//FullSimplify
T2=Table[LF[[i]],[i,0,k-1]]//FullSimplify
f[x_]:=0; F[x_,t_,f1_]:=(t^2*f1[t])/(1+t^2+f1[t]);
F[x_,t_,f1]
Eq1=f[x]-lambda*Integrate[F[x,t,f],{t,a,x}]==g[x]

```

**Problem:** Applying the Laplace transform, solve the Volterra nonlinear homogeneous integral equation of the second kind of convolution type

$$f(x) - \lambda \int_a^x f(t)f(x-t) dt = g(x), \quad g(x) = -x^9, \quad \lambda = \frac{1}{2}, \quad a = 0.$$

Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```

with(inttrans): lambda:=1/2; g:=x->-x^9; a:=0;
Eq1:=0-lambda*Int(f(t)*f(x-t),t=a..x)=g(x);
laplace(g(x),x,p); laplace(f(x),x,p);
Eq1_L:=laplace(Eq1,x,p);

```

```
Sol:=solve(subs(laplace(f(x),x,p)=F(p),Eq1_L),F(p));
f1:=x->invlaplace(expand(Sol[1]),p,x); f1(x);
f2:=x->invlaplace(expand(Sol[2]),p,x); f2(x);
E1:=0-lambda*Int(f1(t)*f1(x-t),t=a..x)=g(x);
E2:=0-lambda*Int(f2(t)*f2(x-t),t=a..x)=g(x);
simplify(value((E1))); simplify(value((E2)));
```

*Mathematica:*

```
g[x_]:=-x^9; {lambda=1/2,a=0,beta=0}
Eq1=beta*f[t]-lambda*Integrate[f[t]*f[x-t],{t,a,x}]==g[x]
{LaplaceTransform[g[x],x,p],LaplaceTransform[f[x],x,p]}
Eq1Lap=LaplaceTransform[Eq1,x,p]/.
{LaplaceTransform[f[x],x,p]->F[p]}
Sol=Solve[Eq1Lap,F[p]]
InvLap1=InverseLaplaceTransform[Sol[[1]],p,x]
f1[t_]:=InvLap1[[1,2]]/.{x->t}; f1[t]
InvLap2=InverseLaplaceTransform[Sol[[2]],p,x]
f2[t_]:=InvLap2[[1,2]]/.{x->t}; f2[t]
E1=beta*f1[t]-lambda*Integrate[f1[t]*f1[x-t],{t,a,x}]==g[x]
E2=beta*f2[t]-lambda*Integrate[f2[t]*f2[x-t],{t,a,x}]==g[x]
```

*Nonlinear integral equations* with constant limits of integration

$$\beta f(x) - \lambda \int_a^b K(x, t, f(t)) dt = g(x).$$

**Problem:** Let  $a = 0$ ,  $b = 1$ . Solve the nonlinear integral equation of the second kind

$$f(x) - \lambda \int_a^b K(x, t) f^2(t) dt = 0, \quad K(x, t) = xt + x^2 t^2,$$

is the degenerate kernel and  $g(x)=0$ . Show that the constructed function  $f(x)$  is the integral equation solution.

*Maple:*

```
lambda:=1; g:=x->0; a:=-1; b:=1;
K:=(x,t)->x*t+x^2*t^2; k:=nops(K(x,t)); var:={C1,C2};
Eq1:=f(x)-lambda*Int(K(x,t)*(f(t))^2,t=a..b)=g(x);
```

```

Eq2:=f(x)-add(lambda*Int(op(i,K(x,t))*(f(t))^2,
t=a..b),i=1..k)=g(x);
Eq3:= isolate(Eq2,f(x));
CC1:= expand(op(1,rhs(Eq3))/x/lambda);
CC2:= expand(op(2,rhs(Eq3))/x^2/lambda);
F := lambda*x*C1+lambda*x^2*C2+g(x);
Eq4:= subsop(2=F, Eq3); f:= unapply(rhs(Eq4),x);
sys := {};
for i from 1 to k do
  CC||i;
  sys:= sys union {value(expand(C||i=CC||i))};
od;
S1:=allvalues({solve(sys, var)});
S2 := S1[1] union S1[2]; m:= nops(S2);
for i from 1 to m do
f||i:=unapply(subs(S2[i],F),x); od;
for i from 1 to m do
simplify(value(f||i(x)
-lambda*Int(K(x,t)*(f||i(t))^2,t=a..b)=g(x))); od;

```

*Mathematica:*

```

g[x_]:=0; K[x_,y_]:=x*y+x^2*y^2
h[u_*v_]:=u*Integrate[v,{y,a,b}]/;FreeQ[u,v]
{lambda=1,a=-1,b=1,k=Length[K[x,y]],
 CA=Array[c,k,1],CC=Array[x1,k,1]}
{Eq1=f[x]-Sum[lambda*h[K[x,y][[i]]*(f[y])^2],
 {i,1,k}]==g[x], Eq2=Solve[Eq1,f[x]]}
CC[[1]]=Eq2[[1,1,2,1]]/x/lambda
CC[[2]]=Eq2[[1,1,2,2]]/x^2/lambda
F=lambda*x*CA[[1]]+lambda*x^2*CA[[2]]+g[x]
f[X_]:=F/.{x->X}; f[x]
sys={}; Do[CC[[i]], sys=Union[sys,{CA[[i]]==CC[[i]]}],
 {i,1,k}]; sys
{S=Solve[sys,CA], m=Length[S], fA=Array[x1,m,1],
 fB=Array[x2,m,1]}
Do[f[X_]:=(F/.S[[i]])/.{x->X}; fA[[i]]=f[x];
 fB[[i]]=f[y],{i,1,m}]; {fA,fB}
Do[Eq1=fA[[i]]-lambda*Integrate[K[x,y]*(fB[[i]])^2,
 {y,a,b}]==g[x]; Print[FullSimplify[Eq1]],{i,1,m}];

```

Singular integral equations  $f(x) - \lambda \int_a^\infty K(x, t)f(t) dt = g(x)$ .

**Problem:** Let  $a = 0$ ,  $b = \infty$ . Solve the singular integral equation with  $K(x, t) = \cos(xt)$  and  $g(x) = 0$ . Show that the constructed functions are the integral equation solutions.

Maple:

```
with(inttrans):
g:=x->0; a:=0; b:=infinity;
K:=(x,t)->cos(x*t); assume(A>0,x>0);
F1:=x->exp(-A*x); F2:=unapply(fouriercos(F1(t),t,x),x);
f1:=x->F1(x)+F2(x); f2:=x->F1(x)-F2(x); f1(x); f2(x);
for i from 1 to 2 do
  In||i :=value(map(Int,expand(f||i(t)*K(x,t)),t=a..b));
  Inn||i:=convert(map(simplify,In||i),exp);
  Eq||i := f||i(x)-lambda*Inn||i=g(x);
  EigenVal||i:=(-1)^(i-1)*sqrt(2/Pi);
  simplify(subs(lambda=EigenVal||i,Eq||i));
  EigenFun||i:=f||i(x);
  simplify(value(f||i(x)
    -EigenVal||i*Int(K(x,t)*f||i(t),t=a..b)=g(x)));
od;
```

Mathematica:

```
{a=0, b=Infinity}
g[x_]:=0; K[x_,t_]:=Cos[x*t];
F1[x_]:=Exp[-A*x];
F2[X_]:=FourierCosTransform[F1[t],t,x]/.{x->X};
{F1[t], F2[t]}
f[x_,m_]:=Module[{},F1[x]+(-1)^(m-1)*F2[x]];
{f[t,1], f[t,2]}
Sol[m_]:=Module[{},
  Int1=Assuming[A>0&&x>0,
  Integrate[f[t,m]*K[x,t],{t,a,b}]]; Print[Int1];
  Eq1=f[x,m]-lambda*Int1==g[x]; Print[Eq1];
  EigenVal=(-1)^(m-1)*Sqrt[2/Pi]; Print[EigenVal];
  Eq1/.{lambda->EigenVal}; Print[Eq1];
  EigenFun=FullSimplify[f[x,m]-EigenVal*Assuming[A>0&&x>0,
    Integrate[K[x,t]*f[t,m],{t,a,b}]]==g[x]]];
{Sol[1], Sol[2]}
```

**Problem:** Let  $a = 0$ ,  $b = \infty$ . Applying the Laplace transform and the Efros theorem on the generalized convolution, solve the singular integral equation with  $K(x, t) = \exp(-t^2/(4x))$  and  $g(x) = \cos x$ . Show that the constructed function is the integral equation solution.

*Maple:*

```
with(inttrans):
g:=x->cos(x); a:=0; b:=infinity;
K:=(x,t)->exp(-t^2/(4*x)); assume(x>0,u>0);
Eq1:=1/(sqrt(Pi*x))*Int(K(x,t)*f(t),t=a..b)=g(x);
Eq2 :=F(sqrt(p))/sqrt(p)=laplace(g(x),x,p);
Eq3:=solve(Eq2,F(sqrt(p)));
Eq4:=simplify(subs(p=u^2,Eq3));
Sol:=unapply(combine(invlaplace(Eq4,u,x)),x);
Eq11:=simplify(value(1/(sqrt(Pi*x))*Int(K(x,t)*Sol(t),
t=a..b)=g(x)));
```

*Mathematica:*

```
g[x_]:=Cos[x]; K[x_,t_]:=Exp[-t^2/(4*x)];
{a=0, b=Infinity}
Eq1=1/(Sqrt [Pi*x])*Integrate[K[x,t]*f[t],{t,a,b}]==g[x]
Eq2=F[Sqrt [p]]/Sqrt [p]==LaplaceTransform[g[x],x,p]
Eq3=Solve[Eq2,F[Sqrt [p]]]/.{p->u^2}
Eq4=Assuming[{u>0,FullSimplify[Eq3]}
Sol[X_]:=FullSimplify[
InverseLaplaceTransform[Eq4,u,x]/.{x->X}];
Sol[x][[1,1,2]]
Eq11=Assuming[x>0,1/(Sqrt [Pi*x])*Integrate[K[x,t]
*Sol[t][[1,1,2]],{t,a,b}]==g[x]]
```

*Systems* of linear integral equations.

**Problem:** Applying the Laplace transform find the exact solution of the system of linear integral equations

$$\begin{aligned} f_1(x) - \lambda_{11} \int_a^x K_{11} f_1(t) dt - \lambda_{12} \int_a^x K_{12} f_2(t) dt &= g_1(x), \\ f_2(x) - \lambda_{21} \int_a^x K_{21} f_1(t) dt - \lambda_{22} \int_0^x K_{22} f_2(t) dt &= g_2(x), \end{aligned}$$

where

$$\begin{aligned} a &= 0, \quad g_1(x) = \sin(x), \quad g_2(x) = \cos(x), \\ K_{11}(x, t) &= K_{12}(x, t) = \cos(x-t), \quad K_{21}(x, t) = K_{22}(x, t) = \sin(x-t), \\ \lambda_{11} &= \lambda_{12} = \lambda_{21} = \lambda_{22} = 1. \end{aligned}$$

Show that the functions  $f_1(x)$ ,  $f_2(x)$  are the solutions of the system of integral equations.

*Maple:*

```
with(inttrans):
lambda11:=1; lambda12:=1; lambda21:=1; lambda22:=1; a:=0;
g1:=x->sin(x); g2:=x->cos(x);
K11:=(x,t)->cos(x-t); K12:=(x,t)->cos(x-t);
K21:=(x,t)->sin(x-t); K22:=(x,t)->sin(x-t);
Eq1:=f1(x)-lambda11*int(K11(x,t)*f1(t),t=a..x)
    -lambda12*int(K12(x,t)*f2(t),t=a..x)=g1(x);
Eq2:=f2(x)-lambda12*int(K12(x,t)*f1(t),t=a..x)
    -lambda22*int(K22(x,t)*f2(t),t=a..x)=g2(x);
sys := {Eq1, Eq2};
sys_L:=laplace(sys,x,p);
Sol:=factor(solve(subs(
    {laplace(f1(x),x,p)=F1(p),
     laplace(f2(x),x,p)=F2(p)}),sys_L,{F1(p),F2(p)}));
F11:=rhs(op(select(has,Sol,F1)));
f1:=x->invlaplace(F11,p,x);
F12:=rhs(op(select(has,Sol,F2)));
f2:=x->invlaplace(F12,p,x);
f1(x); f2(x);
E1:=f1(x)-lambda11*Int(K11(x,t)*f1(t),t=a..x)
    -lambda12*Int(K12(x,t)*f2(t),t=a..x)=g1(x);
E2:=f2(x)-lambda12*Int(K12(x,t)*f1(t),t=a..x)
    -lambda22*Int(K22(x,t)*f2(t),t=a..x)=g2(x);
B1:=simplify(value(E1)); B2:=simplify(value(E2));
```

*Mathematica:*

```
g1[x_]:=Sin[x]; g2[x_]:=Cos[x];
K11[x_,t_]:=Cos[x-t]; K12[x_,t_]:=Cos[x-t];
K21[x_,t_]:=Sin[x-t]; K22[x_,t_]:=Sin[x-t];
{lambda11=1, lambda12=1, lambda21=1, lambda22=1, a=0}
```

```
Eq1=f1[x]-lambda11*Integrate[K11[x,t]*f1[t],{t,a,x}]-  
lambda12*Integrate[K12[x,t]*f2[t],{t,a,x}]==g1[x]  
Eq2=f2[x]-lambda12*Integrate[K12[x,t]*f1[t],{t,a,x}]-  
lambda22*Integrate[K22[x,t]*f2[t],{t,a,x}]==g2[x]  
sys={Eq1,Eq2};  
sysLap=LaplaceTransform[sys,x,p]/.  
{LaplaceTransform[f1[x],x,p]->F1[p]}/.  
{LaplaceTransform[f2[x],x,p]->F2[p]}  
Sol=Solve[sysLap,{F1[p],F2[p]}]  
f1[X_]:=InverseLaplaceTransform[Sol[[1,1,2]],p,x]/.{x->X};  
f2[X_]:=InverseLaplaceTransform[Sol[[1,2,2]],p,x]/.{x->X};  
{f1[x], f2[x],  
Map[FullSimplify,  
{E1=f1[x]-lambda11*Integrate[K11[x,t]*f1[t],{t,a,x}]-  
lambda12*Integrate[K12[x,t]*f2[t],{t,a,x}]==g1[x],  
E2=f2[x]-lambda12*Integrate[K12[x,t]*f1[t],{t,a,x}]-  
lambda22*Integrate[K22[x,t]*f2[t],{t,a,x}]==g2[x]}]}
```

## References

- [1] Abel, M. L., and Braselton, J. P.: Maple V by Example. AP Professional, Boston, MA 1994
- [2] Akritas, A. G.: Elements of Computer Algebra with Applications. Wiley, New York 1989
- [3] Andreev, V. K.: Stability of Unsteady Motions of a Fluid with a Free Boundary. Nauka, Novosibirsk 1992
- [4] Andrew, A. D., and Morley, T. D.: Linear Algebra Projects Using Mathematica. McGraw-Hill, New York 1993
- [5] Aoki, H.: Higher-order Calculation of Finite Periodic Standing Waves by Means of the Computer. *J. Phys. Soc. Jpn.* 49, 1598–1606 (1980)
- [6] Bahder, T. B.: Mathematica for Scientists and Engineers. Addison-Wesley, Redwood City, CA 1995
- [7] Calmet, J., and van Hulzen, J. A.: Computer Algebra Systems. Computer Algebra: Symbolic and Algebraic Computations (Buchberger, B., Collins, G. E., and Loos, R., Eds.), 2d ed. Springer, New York 1983
- [8] Char, B. W., Geddes, K. O., Gonnet, G. H., Monagan, M. B., and Watt, S. M.: Maple Reference Manual, 5th ed. Waterloo Maple Publishing, Waterloo, Ontario, Canada 1990
- [9] Concus, P.: Standing Capillary-gravity Waves of Finite Amplitude. *J. Fluid Mech.* 14, 568–576 (1962)
- [10] Corless, R. M.: Essential Maple. Springer, Berlin 1995

- [11] Davenport, J. H., Siret, Y., and Tournier, E.: Computer Algebra Systems and Algorithms for Algebraic Computation. Academic Press, London 1993
- [12] Heck, A.: Introduction to Maple. Springer, New York 1996
- [13] Geddes, K. O., Czapor, S. R., and Labahn, G.: Algorithms for Computer Algebra. Kluwer Academic Publishers, Boston 1992
- [14] Getz, C., and Helmstedt, J.: Graphics with Mathematica: Fractals, Julia Sets, Patterns and Natural Forms. Elsevier Science & Technology Book, Amsterdam Boston 2004
- [15] Gray, J. W.: Mastering Mathematica: Programming Methods and Applications. Academic Press, San Diego 1994
- [16] Gray, T., and Glynn, J.: Exploring Mathematics with Mathematica: Dialogs Concerning Computers and Mathematics. Addison-Wesley, Reading, MA 1991
- [17] Green, E., Evans, B., and Johnson, J.: Exploring Calculus with Mathematica. Wiley, New York 1994
- [18] Grosheva, M. V., and Efimov, G. B.: On Systems of Symbolic Computations (in Russian). Applied Program Packages. Analytic Transformations (Samarskii, A. A., Ed.). Nauka, Moscow 1988, pp. 30–38
- [19] Krasnov, M., Kiselev, A., and Makarenko, G.: Problems and Exercises in Integral Equations. Mir Publishers, Moscow 1971
- [20] Kreyszig, E., and Normington, E. J.: Maple Computer Manual for Advanced Engineering Mathematics. Wiley, New York 1994
- [21] Maeder, R. E.: Programming in Mathematica, 3rd ed. Addison-Wesley, Reading, MA 1996
- [22] Nayfeh, A.: Perturbation Methods. Wiley, New York 1973
- [23] Okamura, M.: Resonant Standing Waves on Water of Uniform Depth. *J. Phys. Soc. Jpn.* 66, 3801–3808 (1997)
- [24] Penney, W. G., and Price, A. T.: Finite Periodic Stationary Gravity Waves in a Perfect Liquid, Part 2. *Phil. Trans. R. Soc. Lond. A* 224, 254–284 (1952)

- [25] Polyanin, A. D., and Manzhirov, A. V.: *Handbook of Integral Equations*. CRC Press, Boca Raton 1998
- [26] Polyanin, A. D.: *Handbook of Linear Partial Differential Equations for Engineers and Scientists*. Chapman & Hall/CRC Press, Boca Raton 2002
- [27] Ramírez, V., Barrera, D., Posadas, M. y González, P.: *Cálculo numérico con Mathematica*. Ariel Ciencia, Barcelona 2001
- [28] Rayleigh, Lord: Deep Water Waves, Progressive or Stationary, to the Third Order of Approximation. *Phil. Trans. R. Soc. Lond. A* 91, 345–353 (1915)
- [29] Richards, D.: *Advanced Mathematical Methods with Maple*. Cambridge University Press, Cambridge 2002
- [30] Ross, C. C.: *Differential Equations: An Introduction with Mathematica*. Springer, New York 1995
- [31] Sekerzh-Zenkovich, Y. I.: On the Theory of Standing Waves of Finite Amplitude. *Doklady Akad. Nauk. USSR* 58, 551–554 (1947)
- [32] Shingareva, I.: Investigation of Standing Surface Waves in a Fluid of Finite Depth by Computer Algebra Methods. PhD thesis, Institute for Problems in Mechanics, Russian Academy of Sciences, Moscow 1995
- [33] Shingareva, I., and Lizárraga-Celaya, C.: High-order Asymptotic Solutions to Free Standing Water Waves by Computer Algebra. Proc. Maple Summer Workshop (Lopez, R. J., Ed.). Waterloo, Ontario, Canada 2004, pp. 1–28
- [34] Shingareva, I., Lizárraga Celaya, C., and Ochoa Ruiz, A. D.: *Maple y Ondas Estacionarias. Problemas y Soluciones*. Editorial Unison, Universidad de Sonora, Hermosillo, México 2006
- [35] Shingareva, I., and Lizárraga-Celaya, C.: On Frequency-amplitude Dependences for Surface and Internal Standing Waves. *J. Comp. Appl. Math.* 200, 459–470 (2007)
- [36] Schultz, W. W., Vanden-Broeck, J. M., Jiang, L., and Perlin, M.: Highly Nonlinear Standing Water Waves with Small Capillary Effect. *J. Fluid Mech.* 369, 253–272 (1998)

- [37] Tadjbakhsh, I., and Keller, J. B.: Standing Surface Waves of Finite Amplitude. *J. Fluid Mech.* 8, 442–451 (1960)
- [38] Vvedensky, D. D.: Partial Differential Equations with Mathematica. Addison-Wesley, Wokingham 1993
- [39] Whittaker, E. T.: A Treatise on Analytical Dynamics of Particles and Rigid Bodies. Dover Publications, New York 1944
- [40] Wickam-Jones, T.: Mathematica Graphics: Techniques and Applications. Springer, New York 1994
- [41] Wester, M. J.: Computer Algebra Systems: A Practical Guide. Wiley, Chichester, UK 1999
- [42] Wolfram, S.: A New Kind of Science. Wolfram Media, Champaign, IL 2002
- [43] Wolfram, S.: The Mathematica Book, 5th ed. Wolfram Media, Champaign, IL 2003
- [44] Yosihara, H.: Gravity Waves on the Free Surface of an Incompressible Perfect Fluid of Finite Depth. *Kyoto Univ. Math.* 18, 49–96 (1982)
- [45] Zimmerman, R. L., and Olness, F.: Mathematica for Physicists. Addison-Wesley, Reading, MA 1995

# General Index

- abbreviation, 8
- arc length, 119
- Archimedes's spiral, 98
- asymptotic solution, 193, 195
- averaging transformations, 185
- axis of rotation, 104
- basis of
  - column space, 76, 77
  - nullspace, 76, 77
  - row space, 76, 77
- Bessel functions, 151, 162
- Beta function, 148
- binary
  - hardware system, 7
  - number system, 7
- bounded region
  - for graphs, 55
  - volume of, 132
- Cauchy principal value, 206
- Cauchy's integral theorem, 144
- Cayley-Hamilton theorem, 83, 84
- characteristic
  - matrix, 83, 84
  - polynomial, 83, 84
- Chebyshev polynomials, 149
- Christophel symbol, 89, 90
- circular membrane, 170
- commutator relation, 88
- complex
  - algebra, 135
  - argument, 135, 136
  - conjugate, 135, 136
  - number, 135, 136
  - absolute value of, 135, 136
  - evaluation, 135, 136
  - imaginary part of, 135, 136
  - limit of, 139
  - polar representation of, 135
  - real part of, 135, 136
  - sign of, 135, 136
- plane, 141
- variable, 135, 136
- computation domain
  - extension of, 68
- computation time, 16, 32, 34, 35
- conformal mapping, 143, 144
- constants, 32
  - mathematical, 10, 32
  - predefined, 10
- continuity equation, 194, 197
- convolution theorem, 210, 228
- coordinate
  - basis, 90
  - lines, 54
- coordinates
  - cylindrical, 106
  - polar, 62, 97
  - spherical, 106, 107
- coprimes, 97
- curves
  - animations of, 61
  - area between, 98, 119
  - implicitly defined, 96, 103
  - in polar coordinates, 96

- d'Alembert's solution, 163, 164  
data  
    structure, 19  
    type, 9, 30  
De Moivre's theorem, 136, 137  
decimal number system, 7, 26  
decomposition  
    LU, 82  
    polar, 82  
degenerate kernel, 220, 225, 228  
degree of freedom, 15, 37  
density plot, 58  
derivative  
    covariant, 89  
    directional, 89  
    partial, 127–129  
differential  
    calculus, 109–114  
    operator, 195  
Dirac delta function, 154  
discrete  
    Fourier transform, 164, 168  
    transform, 155–168  
dispersion relation, 194  
distribution functions, 154  
divergence to infinity, 141  
division algorithm, 34  
double  
    factorial, 17, 38  
    pendulum, 15, 37  
    quotes, 30, 31  
dynamical system, 141, 185
- Efros theorem, 228  
eigenvalue, 83, 84  
eigenvector, 83, 84  
elimination  
    fraction-free, 81, 82  
    methods, 81, 82  
    reduced row-echelon, 81, 82  
ellipsoid, 102  
environment variable, 7, 120
- equation  
    algebraic, 171–176  
    ordinary differential, 151, 177  
    parametric, 97  
    partial differential, 187–204  
    real roots of, 13  
    trascendental, 150
- error  
    approximation, 124  
    function, 148, 149  
    upper bound of, 123
- Euclidean space, 91, 205  
Euler's  
    constant, 5, 23  
    equations, 192  
Eulerian variables, 192  
exponential integral, 162  
expression, 9, 26, 30  
    boolean, 9, 30  
    common factors of, 69  
    converting, 68  
    denominator of, 68, 69  
    evaluation of, 69  
    head of, 26  
    numerator of, 68, 69  
    simplification of, 69  
    trigonometric, 71  
    unvaluated form of, 69
- external flow, 134
- factorial, 34, 38  
fast Fourier transform, 164, 166  
Fermat's spiral, 98  
Fibonacci numbers, 16, 34, 35  
finite difference method, 191  
fluid motion, 185  
Fourier  
    coefficient, 124, 125, 168  
    polynomial, 167  
    series, 124, 126  
    transform, 155, 161, 162, 168
- fractals, 141

- Fredholm  
determinant method, 214, 216  
integral equation, 206–221
- free  
boundary problem, 202  
surface, 192  
vibration, 170
- frequency  
amplitude dependence, 192  
correction, 185  
spectrum, 165, 167
- Fresnel integral, 148, 149
- function  
absolute value, 34  
active, 11  
analytic, 138  
animation of, 20  
application, 31  
arithmetic, 27  
composition, 12, 13, 34, 35  
definitions of, 12  
derivative, 109, 110  
digamma, 148  
elementary trascendental, 11  
elliptic, 149  
evaluation of, 12, 34  
exponential, 33  
harmonic, 138  
holomorphic, 144  
hyperbolic, 11, 33  
hypergeometric, 153, 154  
inert, 11  
library, 11  
limit of, 109, 110  
multivalued, 137, 138  
multivariable, 12, 120, 127, 128  
periodic, 126  
periodic extension of, 14, 35  
piecewise continuous, 13, 35  
polygamma, 148  
predefined, 11, 32  
pure, 32, 178
- random, 34  
recursive, 35  
rose, 98  
sign, 34  
signum, 135, 136  
special, 11, 32  
specialized, 25  
square integrable, 212  
trascendental, 33  
trigonometric, 11, 33  
user-defined, 11, 12, 32, 34  
vector, 12, 34
- Gamma function, 148
- Gauss-Jordan elimination, 81, 82
- Gaussian elimination, 81, 82
- Gegenbauer polynomials, 149
- general relativity theory, 85, 86, 90
- generalized  
coordinates, 15  
forces, 15
- geometric shape, 107, 108
- gradient field, 105
- Gram-Schmidt  
orthonormalization, 82, 83
- graphs  
area between, 98  
bar, 59, 60  
intersection of, 92, 175  
logarithmic, 56  
multiple, 50–51  
of function, 47  
of piecewise  
continuous function, 57  
options of, 48, 50  
simple, 47  
special, 54–58  
structures of, 51  
text strings in, 53  
with color gradient, 58  
with legend, 58
- greatest common divisor, 34, 66

- Hankel  
functions, 151  
matrix, 78  
transform, 168
- hardware precision, 164
- harmonic conjugate, 138, 139
- Heaviside unit step function, 154
- Helmholtz's equation, 168
- Hermite's polynomials, 149
- Hermitian  
antisymmetric matrix, 88  
transpose matrix, 77, 78  
transpose vector, 77
- Hilbert  
matrix, 78  
transform, 155
- hyperbolic spiral, 98
- hyperboloid  
of one sheet, 102  
of two sheets, 102
- inequality, 9, 56
- inner product, 82, 83
- integral  
approximation, 115  
around poles, 145  
by parts, 115, 118  
by substitution, 115, 117  
definite, 115  
elliptic, 149  
exponential, 148  
indefinite, 115  
iterative, 127  
logarithmic, 148  
multiple, 127, 128  
polylogarithmic, 148  
trigonometric, 148
- integral calculus, 115–120
- integral equation, 204–230  
eigenfunction of, 221  
eigenvalue of, 221  
exact solution, 206
- kernel of, 205
- linear, 205
- nonlinear, 205, 223–225
- of convolution type, 207, 224
- of the first kind, 205
- of the second kind, 205
- of the third kind, 205
- singular, 206, 227, 228
- with constant  
limits of integration, 205
- with variable  
integration limit, 205
- integral transform, 155–164
- interface parametrization, 202
- inverse  
function, 172  
hyperbolic function, 33  
trigonometric function, 33
- Jacobi polynomials, 149
- Jacobian, 193
- Jordan  
block, 84, 85  
canonical form, 84  
decomposition, 85  
transformation matrix, 84
- Julia set, 141, 142
- kinetic energy, 15
- knots, 99, 100
- Kronecker product, 87
- Lagrange  
equations, 15, 37, 194  
multipliers, 131
- Lagrangian, 15  
formulation, 192  
mechanics, 15  
variables, 192
- Laguerre polynomials, 149
- Lambert function, 150
- Laplace transform, 155, 157, 207, 210, 211, 224, 228, 229

- Laplace's equation, 168
- Laurent series, 145
- least common multiple, 34
- Legendre
  - functions, 153
  - polynomials, 33, 149
- level
  - curves, 102, 103
  - surfaces, 102, 103
- Levi-Civita
  - symbol, 88
  - tensor, 88
- Lie product, 88
- linear
  - algebra, 71, 100
  - combination, 77
  - dependence, 77
  - equation, 79, 81
  - space, 71
  - system, 79, 81
- Lissajous curves, 62, 97
- lists, 39, 42
  - empty, 39
  - nested, 39, 40, 86
  - simple, 39, 40
  - sublists of, 42
- logarithm
  - general, 33
  - natural, 33
- Möbius strip, 25
- machine code, 26
- Maclaurin
  - polynomial, 122, 123
  - series, 120, 122, 123
- Mandelbrot set, 141, 143
- map of North America, 25
- Mathieu
  - equation, 52
  - functions, 152
- matrix, 43
  - adjoint, 79
- algebra, 74
- augmented, 79, 81
- band, 77, 78
- basis of, 77
- block diagonal, 84, 85
- coefficient, 79, 81
- cofactor, 79
- components, 72, 73
- constant, 77, 78
- corank of, 79, 81
- determinant, 6, 79
- diagonal, 77, 78
- diagonalization of, 83, 84
- dimension, 76, 77
- identity, 77, 78
- inverse, 79
- kernel of, 76, 77, 79, 81
- minors of, 79
- random, 6, 77, 78
- rank of, 79, 81
- representation, 72, 73
- sparse, 77, 78
- special, 77, 78
- trace of, 83, 84
- transposition, 77, 78
- visualization, 75, 76
- zero, 77, 78
- maximum, 11, 36, 131
  - amplitude, 204
  - wave profile, 204
- Mellin transform, 155
- memory option, 35
- minimal
  - distance, 113
  - polynomial, 83, 84
  - volume, 114
- minimum, 11, 131
- module, 14, 16, 36, 37
- multivariate calculus, 127–134
- Newton's method, 173
- nonlinear frequency, 194

- notation  
engineering, 26  
mathematical, 86  
scientific, 26  
tensor, 86  
number ring, 65  
numerical  
approximation, 7, 26, 168, 174  
solution, 173–175
- ODE, 157, 177–186  
analytical solution, 177, 178  
animations, 181  
boundary value  
problem for, 182  
direction field, 183  
exact solution, 177, 179, 180  
explicit form of solution, 178  
general solution, 178  
graphical solution, 179–182  
higher order, 179  
homogeneous, 178  
implicit form of solution, 178  
initial value problem  
for, 157, 159, 160, 182, 184, 210  
integral transform  
method, 177  
interactive  
numeric solving, 177  
symbolic solving, 177  
linear, 177, 178, 208  
numerical solution, 179, 181  
phase portrait, 181, 185, 186  
separation of variables, 178  
series solution, 177  
solution classification, 178  
solution suggestion, 178  
system, 160, 177, 183  
vector field, 181  
with polynomial  
coefficients, 177
- operator  
arithmetic, 8, 27  
assignment, 8, 28  
composition, 12  
conditional, 35  
logic, 8, 9, 27, 30  
relation, 8, 9, 27, 30  
replacement, 34  
unassignment, 8, 29
- orthogonal  
functions, 149  
kernels, 218  
matrices, 82, 83  
polynomials, 149  
vectors, 82, 83
- orthogonality condition, 198  
orthonormal vectors, 82, 83
- parametric curve, 94, 95, 193  
pattern, 28, 34, 38, 41  
matching, 28, 68
- PDE, 163, 164, 170, 187–191  
analytical solution, 187  
boundary value problem  
for, 190  
general solution, 187  
graphical solution, 189, 190  
initial value problem  
for, 163, 164, 170, 188  
initial/boundary value problem  
for, 158, 191  
linear system, 196  
method of characteristics, 188  
numerical solution, 187, 189  
separation of variables, 187  
system, 187
- perturbation theory, 176, 195  
phase space, 141  
point  
along a curve, 64  
critical, 112, 130  
inflection, 112

- polynomial
  - coefficients, 65, 66
  - definition, 66, 67
  - degree, 65, 66
  - distribution, 68, 69
  - division, 65, 66
  - equations, 66
  - factorization, 68, 69
  - multivariate, 65
  - normal form of, 68, 69
  - roots, 65, 66, 172
  - simplification, 68, 69
  - sorting, 65
  - type, 65, 66
  - univariate, 65
- potential energy, 15
- power series, 176, 195
- precision
  - arbitrary, 31
  - machine, 31
  - user-specified, 7, 26
  - with accuracy, 31
- previous results, 10, 31
- prime number, 34
- problem
  - ill-posed, 212
  - well-posed, 202
- procedure, 14
  - recursive, 16
- radius of convergence, 140, 141
- recurrence relation, 216
- regularization method, 212
- relative
  - extremum, 127, 128
  - minimum/maximum, 115
- removable discontinuity, 139
- repetition structure, 37
- resolvent kernel, 210, 214, 216–218
- resonance parameter, 185
- Riemann
  - curvature tensor, 89, 90
- sphere, 141
- sum, 115, 116
- Zeta function, 153, 154
- Schwarzschild metric, 90
- secant line, 99
- sequence
  - animations of, 62, 63
  - graph of, 20, 44, 55
  - of arguments, 14
  - of characters, 9, 30
- series, 120–127
  - coefficient, 120, 121
  - convergence, 121, 122, 140
  - expansion, 120, 121
  - infinite, 153, 154
  - inverse, 121
  - inversion of, 70
  - multiplication, 120
  - polynomial
    - representation of, 120, 121
  - sum, 120, 121
- sign condition, 202
- signal recuperation, 165, 166
- significant digits, 7, 26
- simplification
  - algebraic, 68
  - rules, 69
  - trigonometric, 70
  - using assumptions, 69
- Simpson’s rule, 118
- singularity, 145
  - limit at, 146
- small parameter, 176, 185, 194
- solid of revolution
  - volume of, 119, 120
- solution
  - approximate, 79, 81
  - analytical, 176, 192
  - by least squares, 81
  - exact, 79, 81, 171, 172
  - periodic, 192

- special functions, 147–154
- special options, 14
- stability diagram, 52
- statement, 8, 14
  - separators, 9
- string, 9
  - fixed, 191
  - semi-infinite, 158
- strings, 30
- structure
  - animations of, 64
  - applying a function to, 20
  - control, 17, 37
  - dimension of, 40
  - element of, 19
  - empty, 19
  - inserting, 20
  - length of, 40
  - number of elements, 19
  - properties, 20
  - rank of, 40
  - repetition, 17
  - replacing, 20
  - selection, 17, 37
  - substructure of, 19, 20, 41
- structures
  - concatenation of, 19
  - difference of, 20
  - division of, 20
  - intersection of, 20
  - multiplication of, 20
  - removing, 20
  - sum of, 20
  - union of, 20
- subharmonic resonance, 185
- successive approximations
  - method, 212, 223
- sum of partial fractions, 69
- surface
  - implicitly defined, 101
  - of revolution, 104
  - parametrically defined, 101
- tension, 202
- symbol, 26
- system
  - holonomic, 15, 37
  - of equations, 171–173, 175
  - of integral equations, 229
- table, 43
- tangent
  - line, 99, 111, 113
  - plane, 130
- Taylor's
  - polynomial, 120, 124
  - series, 120, 122, 123
- tensor, 43
  - algebra, 87
  - analysis, 86
  - antisymmetric, 86, 88
  - calculus, 85, 86
  - components, 85, 86
  - contravariant metric, 85, 86
  - converting, 85, 86
  - covariant metric, 85, 86
  - curvature, 90
  - definition of, 85, 86
  - differentiation, 85, 86, 89
  - index character, 85, 86
  - operation of, 85, 86
  - product, 87
  - rank, 85, 86
  - subtensor of, 86
  - transposition, 87
  - type, 85
  - visualization, 86
- tensors
  - linear combination of, 87
  - operations of, 85, 86
- Tikhonov's
  - regularization method, 213
- total
  - derivative, 128, 129
  - differential, 128, 129

- transformation rules, 26, 172
- transverse vibration, 158
- trapezoidal rule, 118
- trascendental equation, 171–173
- tree structure, 9, 30
- tubes, 99, 100
- type
  - of brackets, 9, 31
  - of numbers, 10, 32
  - of quotes, 10, 31
- range, 9
- unevaluated form, 7, 26
- unit normal, 202
- units of angle, 10
- variable
  - global, 14, 36
  - local, 14, 36
  - with assigned value, 8
  - with no assigned value, 8
- vector
  - algebra, 73, 74
  - components, 71, 72
  - constant, 77
  - field, 105, 134
  - random, 77
  - space, 76, 77, 82, 83
  - sparse, 77
  - unit, 77, 133
  - visualization, 75, 76
  - zero, 77
- vectors
  - angle between, 82, 83
  - representation of, 71, 72
  - special, 77
  - transposition of, 77
- Volterra
  - integral equation, 206–224
- wave equation, 62, 158, 187, 190
- wave number, 194
- waveform, 126, 127
- waves
  - standing, 192
  - traveling, 61, 62

## *Maple* Index

- ' , 57, 66, 126, 131, 139, 142
- ' ', 8, 10
- ( ), 9, 12
- >, 12, 20, 79, 84, 127
- ., 80, 127, 222
- ., ., 9, 13, 18
- ::, 9, 190
- :-, 16, 189
- :::, 16, 169, 201
- :=, 8, 20
- ;, 9
- <, 8, 71–74, 76, 77, 80–83, 127
- <>, 18, 93
- <=, 8, 13
- <>, 8
- =, 8
- >, 5, 8
- >=, 8, 13
- ? , 5
- @ , 12, 13, 53, 109, 118
- [ ], 9, 12, 18–20, 50, 63, 71–73, 75, 80, 112, 114, 119, 121–123, 125, 130–132
- # , 10, 16, 20
- \$ , 130, 131, 137, 149, 180, 187
- % , 10
- &\*, 20, 79, 83, 84, 183
- &x , 127
- , 8
- { }, 9, 18, 19, 50, 76, 173, 180, 190
- ' , 130, 131
- ' ', 10, 15
- Add, 73, 74
- Adjoint, 79
- BandMatrix, 78
- Basis, 76
- BesselI, 151
- BesselJZeros, 151
- BesselJ, 151
- BesselK, 151
- BesselYZeros, 151
- BesselY, 151, 162
- Beta, 148
- Catalan, 10
- Characteristic  
    Matrix, 83
- Polynomial, 83
- Christoffel1, 89
- Christoffel2, 89
- Ci, 148
- ColumnDimension, 76
- ColumnOperation, 75
- ColumnSpace, 76
- Complex, 10
- ConstantMatrix, 78
- ConstantVector, 77
- CrossProduct, 127
- Curl, 127, 133
- DEplot, 181, 183
- DeleteColumn, 75
- DeleteRow, 75
- Determinant, 79, 215, 220, 222
- DiagonalMatrix, 78, 84
- Diff, 11, 109, 127, 144, 154

Digits, 7, 164, 167, 204  
Dimension, 76  
Dirac, 154, 155, 169  
Divergence, 127, 132, 133  
DotProduct, 82, 127  
Doubleint, 127  
Determinant, 6  
D, 128, 158, 163, 180, 184, 185  
Eigenvalues, 83  
Eigenvectors, 83  
Ei, 148, 162  
EllipticE, 149  
EllipticF, 149  
EllipticK, 149  
EllipticPi, 149  
EqLagrange, 15  
FAIL, 10  
FFT, 164, 166  
Fib, 16  
Float, 10  
Format, 6  
FourierTransform, 165, 167  
FresnelC, 148  
FresnelS, 148  
FunctionAdvisor, 11, 147  
GAMMA, 148  
GaussianElimination, 81  
GenerateEquations, 80  
GenerateMatrix, 80, 220, 222  
    augmented, 80  
Gradient, 127, 132, 133  
GramSchmidt, 82  
    normalized, 82  
HankelH1, 151  
HankelH2, 151  
Heaviside, 8, 126, 154, 155, 159,  
    169  
Help menu, 5  
HermiteH, 149  
HermitianTranspose, 77, 78  
IdentityMatrix, 78, 79, 81  
Im, 136, 167  
Insert, 6  
IntersectionBasis, 76  
Int, 11, 117–119, 132, 144, 154, 215  
    AllSolutions, 214  
InverseFourier  
    Transform, 165  
IsOrthogonal, 82  
IsUnitary, 82  
I, 10, 135  
Jacobian, 127  
JordanBlock, 84  
JordanForm, 84  
JuliaSet2D, 142  
JuliaSet3D, 142  
L\_tan , 113  
Laplacian, 127, 132, 133, 190  
LegendreP, 153  
LegendreQ, 153  
Limit, 11  
LinearSolve, 80, 220  
Li, 148  
MandelbrotSet, 142  
MathieuA, 52, 152  
MathieuB, 52, 152  
MathieuCEPrime, 152  
MathieuCE, 152  
MathieuC, 152  
MathieuExponent, 152  
MathieuFloquet, 152  
MathieuSE, 152  
MathieuS, 152  
MatrixAdd, 74  
MatrixMatrixMultiply, 74  
MatrixNorm, 82  
MatrixScalarMultiply, 74  
MatrixVectorMultiply, 74  
Matrix, 18, 72, 74, 75, 78–80, 83,  
    84, 215  
    shape, 78  
MaxMin, 16  
Maximum, 14  
MinimalPolynomial, 83

Minor, 79  
Multiply, 74  
NGSol, 184  
NULL, 16, 19, 20, 64, 167, 182, 184, 210, 222  
Nabla, 127  
Normalize, 82  
Norm, 82  
NullSpace, 76  
ON, 158, 163, 170, 187  
Order, 120  
PieChart, 59  
Pi, 10  
Psi, 148  
RETURN, 15  
RandomMatrix, 78  
    outputoptions, 78  
RandomVector[column], 77  
Range, 169  
Rank, 81  
ReducedRowEchelonForm, 81  
RegPertPoly, 176  
Re, 136, 165, 167  
RiemannSum  
    method, 116  
    partition, 116  
Riemann, 89  
RootOf, 10, 171  
RowDimension, 76  
RowOperation, 75  
RowSpace, 76  
ScalarMatrix, 78  
ScalarMultiply, 75  
ScalarVector, 77  
ScatterPlot, 91, 93  
SetCoordinates, 133  
Si, 148  
SubMatrix, 75  
SumBasis, 76  
Tangent, 99  
Transpose, 77–79  
Tripleint, 127  
UnitVector, 77  
VConj, 138  
VectorAdd, 73  
VectorAngle, 82  
VectorField, 127, 133  
VectorNorm, 82  
Vector, 18, 71, 82, 165, 167, 182  
    row, 71  
View, 6  
ZeroMatrix, 78  
ZeroVector, 77  
Zeta, 153  
\_C1, 178, 183  
about, 136  
abs, 122, 136, 142, 162  
addtable, 169  
add, 15, 66, 126, 167, 176, 210, 215, 220  
alias, 8, 150  
allvalues, 130, 131, 137, 171, 226  
and, 8, 13  
angle, 10, 82  
animate3d, 61, 189  
animatecurve, 61  
    coords, 62  
    frames, 62  
    thickness, 62  
animate, 61, 186, 189, 190, 204  
    background, 64  
    color, 61  
    frames, 61, 99  
    view, 99  
arccosh, 11  
arccos, 11  
arccoth, 11  
arccot, 11  
arccsch, 11  
arccsc, 11  
arcsech, 11  
arcsec, 11  
arcsinh, 11  
arcsin, 11

arctanh, 11  
arctan, 11  
argument, 136  
array, 18–20, 61, 66, 72, 85, 87, 97,  
    130, 131, 165, 167, 180, 183  
    sparse, 77, 78, 85, 89, 90  
    symmetric, 85, 89, 90  
assign, 8, 160, 180, 183  
assume, 68, 136, 155, 159, 169, 227  
assuming, 214  
asympt, 151  
augment, 80, 81  
backsub, 81  
basis, 76  
break, 8, 17, 18  
by, 8  
casesplit, 187  
cat, 15, 19, 66  
ceil, 11  
changevar, 115, 117  
charmat, 83  
charpoly, 83  
circle, 107  
coeffs, 65  
coeff, 65, 125, 176  
coeftayl, 120  
coldim, 80  
collect, 65, 125, 176  
combine, 68, 118, 133, 159  
    trig, 12, 70, 125  
cone, 107  
conformal3d  
    grid, 143  
    orientation, 143  
conformal, 54, 139, 151  
    grid, 54, 143  
    numxy, 143  
conjugate, 136  
constants, 10  
contourplot3d  
    orientation, 103  
contourplot, 178  
coloring, 103  
contours, 103  
filled, 103  
grid, 103, 105  
convert, 9, 10, 68  
    +, 201  
    Bessel, 151  
    Ei, 148  
    GAMMA, 148  
    Heaviside, 154, 162  
    Standard  
        Functions, 140, 153  
    ‘+‘, 66  
    exp, 163  
    factorial, 148  
    hypergeom, 153  
    int, 163, 170  
    listlist, 85  
    list, 15, 130, 131, 176, 201, 220  
    matrix, 73, 83  
    parfrac, 68  
    piecewise, 154, 158, 159  
    polar, 136  
    polynom, 70, 120, 122, 123  
    set, 125  
    string, 97  
    vector, 73  
coordplot, 54  
    cartesian, 54  
    grid, 54  
    view, 54  
corank, 80  
cosh, 11  
cos, 11  
coth, 11  
cot, 11  
cov\_diff, 89  
create, 85, 87, 89, 90  
csch, 11  
csc, 11  
csgn, 136  
cuboid, 107

cylinderplot  
    grid, 106  
cylinder, 107  
dimetric, 89  
d2metric, 89  
declare, 158, 163, 170, 187  
defform, 129  
degrees, 10  
degree, 65  
denom, 68  
densityplot, 58, 142  
    axes, 58  
    colorstyle, 58  
    numpoints, 58  
    style, 58  
det, 6, 79  
diagonal, 20  
diag, 78  
diff, 11, 15, 109, 123, 127, 128,  
    138, 144, 150, 154  
dilog, 148  
directional\_diff, 89  
display3d, 51, 95, 131, 132  
    insequence, 61, 63, 64  
    orientation, 53, 64  
    scaling, 64  
display\_allGR, 90  
display, 51, 53, 55, 56, 61, 92, 97,  
    130, 165, 167, 180  
    axes, 54  
    insequence, 20, 61, 62, 97  
    scaling, 54, 58  
    title, 51  
    xtickmarks, 54  
    ytickmarks, 54  
distance, 113  
diverge, 134  
divide, 65  
dotprod, 133  
do, 8  
dsolve, 159, 163, 180, 182, 183, 208  
    coeffs, 177  
formal\_series, 177  
implicit, 178  
interactive, 177  
method, 160, 177  
numeric, 180, 181  
output, 181, 184  
range, 184  
series, 177, 178  
type, 184  
d, 129  
eigenvals, 83  
eigenvectors, 183  
eigenvects, 83  
eliminate, 143  
end, 8  
entermatrix, 72  
entermetric, 85  
environment, 7  
equal, 73, 79, 81  
equate, 183  
erfc, 148  
erf, 148, 155  
evala, 68, 158  
evalb, 138  
evalc, 11, 78, 136, 137, 140, 165  
evalf, 5, 7, 11, 12, 77, 121, 123,  
    126, 130, 131, 173, 174  
evalhf, 7  
evalm, 20, 73–75, 77–79, 83, 84  
evaln, 8  
eval, 12, 74, 85, 90, 92, 123, 159  
exp(1), 10  
expand, 5, 11, 68, 70, 83, 136  
exp, 11  
extrema, 127, 131  
factorial, 215  
factor, 68, 74, 79, 133  
false, 10  
ffgausselim, 81  
fieldplot3d  
    arrows, 105  
    grid, 105

fieldplot, 182  
    arrows, 105  
    grid, 105  
floor, 11  
for, 8, 13, 15–18, 20, 62–64, 66, 76,  
    83, 87, 92, 97, 142, 176, 184,  
    191, 210, 215  
    while, 17, 18, 142  
fouriercos, 161, 162, 227  
fouriersin, 161, 162  
fourier, 161, 163  
frac, 11, 14  
frontend, 68  
fsolve, 55, 67, 92, 119, 173  
    ..., 173, 174  
    avoid, 174  
    complex, 173, 174  
function, 8  
gamma, 5, 10  
gausselim, 81  
gaussjord, 81  
genhomosol, 178  
get\_char, 85  
get\_compts, 85, 90  
get\_rank, 85  
gradplot3d  
    arrows, 105  
    grid, 105  
gradplot  
    arrows, 105  
    grid, 105  
grad, 132, 133  
hankel, 169, 170  
has, 200  
help, 5  
histogram, 59  
hypergeom, 153  
iFFT, 164, 166  
identity, 20  
if, 8, 13, 14, 16–18, 57, 114, 118,  
    126, 139, 142  
implicitplot3d, 103  
    grid, 103  
implies, 8  
index[function], 147  
index[package], 6  
inequal, 55, 56  
    optionsexcluded, 56  
    optionsfeasible, 56  
    optionsopen, 56  
infinity, 10, 227  
inifcn, 147  
inifncs, 8  
ininames, 8  
ininame, 10  
interface, 10  
intersect, 20  
intparts, 115, 118  
introduction, 5  
int, 11, 126, 127, 134, 138, 154, 199  
inverse, 79, 83, 120  
invert, 85, 89  
invfourier, 161–163  
invlaplace, 160, 225, 228, 229  
isolate, 114, 171, 206, 208, 220,  
    226  
ithprime, 97  
jordan, 84  
kernel, 80  
laplace, 155, 158–160, 180, 206,  
    225, 229  
laurent, 145  
lcoeff, 65  
ldegree, 65  
leastsqrs, 80  
leftbox, 115  
leftsum, 115  
lhs, 8, 9, 178  
limit, 11, 109, 121, 122, 139, 158,  
    162  
    complex, 146  
lin\_com, 87  
line, 107

**linsolve**, 80  
  **ZeroVector**, 80  
**listplot3d**, 91, 93  
**listplot**, 91, 93, 165  
**ln**, 11  
**local**, 14  
**log10**, 11  
**log[b]**, 11  
**loglogplot**, 56  
**logplot**, 48, 56  
  **color**, 56  
  **style**, 56  
**map**, 20, 68, 74, 78, 79, 83–85, 125,  
  130, 133, 174, 227  
**matadd**, 73  
**match**, 68  
**matrixDE**, 180, 183  
**matrixplot**, 91  
  **gap**, 75  
  **heights**, 75  
  **style**, 75  
**matrix**, 18–20, 72, 74, 84  
**max**, 11  
**minpoly**, 83  
**minus**, 20  
**min**, 11  
**modp**, 17  
**module**, 16  
  **export**, 16  
**mod**, 14  
**mtaylor**, 120  
**multiply**, 74, 81, 120  
**names**, 8  
**next**, 17  
**nops**, 9, 15, 19, 76, 92, 118, 130,  
  131, 176, 184, 201, 220  
**normal**, 15, 68  
**not**, 8  
**numer**, 68, 113  
**odeadvisor**, 178  
**odeplot**, 48, 180, 181, 184  
**odetest**, 160  
  **op**, 9, 19, 20, 62, 63, 114, 118, 123,  
    130–132, 163, 167, 201  
  **orthopoly[G]**, 149  
  **orthopoly[H]**, 149  
  **orthopoly[L]**, 149  
  **orthopoly[P]**, 149  
  **orthopoly[T]**, 149  
  **orthopoly[U]**, 149  
  **or**, 8  
  **package**, 6  
  **partial\_diff**, 89  
**pdsolve**  
  **HINT**, 187  
  **build**, 187  
  **numeric**, 189, 190  
**phaseportrait**, 181, 185, 186  
**piecewise**, 13, 57, 155, 162, 214,  
  222  
**plot3d[options]**, 48  
**plot3d**, 5, 47, 48, 95, 189, 190  
  **grid**, 48, 53, 103, 119, 128  
  **orientation**, 102, 103, 128  
  **shading**, 128, 130  
  **style**, 48, 64  
**plot[options]**, 48, 55  
**plots**  
  **arrow**, 10  
**plot**, 5, 13, 20, 47, 50, 189  
  **color**, 50  
  **coords**, 96  
  **discont**, 48, 54, 126  
  **filled**, 55  
  **gridlines**, 54  
  **linestyle**, 50, 150  
  **numpoints**, 13  
  **scaling**, 48  
  **style**, 20, 48, 62, 121, 191  
  **symbolsize**, 20  
  **symbol**, 20, 62, 121, 191  
  **thickness**, 54  
  **tickmarks**, 48  
  **title**, 48

pointplot3d, 91, 93  
  axes, 92  
  symbolsize, 92  
  symbol, 75, 92  
pointplot, 64, 91, 93, 142, 165  
  symbol, 75, 92, 112  
polarplot, 96, 97  
  filled, 134  
polar, 136  
polynomials, 65  
printf, 9, 17, 18, 191  
print, 10, 13, 71–73, 125, 137, 218  
procedure, 14  
proc, 8, 12–17, 57, 113, 123, 126,  
  142, 199, 201  
  args, 14  
  global, 14  
  local, 14  
  option remember, 16  
  options, 14  
prod, 87  
radians, 10  
randmatrix, 6  
random, 56  
rand, 77  
rank, 80  
remove  
  has, 20, 114, 118, 199  
reserved, 8  
residue, 145  
restart, 8  
return, 8  
rhs, 8, 9, 158, 163, 170, 180, 184  
rightbox, 115  
rightsum, 115  
roots, 65  
rotate, 107  
round, 11  
save, 8  
scalarmul, 133  
scale, 107  
sech, 11  
sec, 11  
selectSol, 67  
select  
  has, 19, 114, 199, 229  
  type, 123  
semilogplot, 56  
separability, 187  
separablesol, 178  
seq, 18, 20, 56, 58, 63, 93, 121, 130,  
  131, 165, 191  
series, 11, 120, 122, 123, 145, 150,  
  152  
setoptions3d, 50, 130, 132  
setoptions, 50, 61, 119, 134  
shortcut, 6  
signum, 136  
simplify, 11, 12, 68, 79, 83, 84  
  HermiteH, 149  
  LegendreP, 153  
  trig, 70  
simpson, 115, 118  
singular, 145  
sinh, 11  
sin, 11  
solve, 5, 13, 67, 70, 123, 130, 132,  
  137, 143, 150, 157–159, 173  
sort, 5, 65, 176  
spacecurve, 95, 131  
  axes, 63  
  shading, 63  
  style, 63  
sparse, 20  
sphereplot  
  grid, 106  
  orientation, 106  
sphere, 107  
sprintf, 58  
statplots, 59  
submatrix, 81  
subsop, 20, 163, 220, 226  
subs, 12, 20, 68, 83, 113, 114, 123,  
  144, 163, 170, 178

**sum**, 66, 120, 121, 126, 140, 153,  
209, 218  
**symmetric**, 20  
**table**, 18, 19  
**tanh**, 11  
**tan**, 11  
**taylor**, 120  
**tcoeff**, 65  
**tensorsGR**, 90  
**testeq**, 70, 71  
**textplot3d**, 53  
    **color**, 53  
    **font**, 53  
**textplot**, 53, 58  
    **color**, 53  
    **font**, 53  
**time**, 16  
**trace**, 83  
**translate**, 107  
**trapezoid**, 115, 118  
**true**, 10  
**trunc**, 11, 14  
**tubeplot**  
    **numpoints**, 99  
    **radius**, 99  
    **tubepoints**, 99  
**type**, 9, 85  
     $+$ , 199  
    **function**, 118  
    **matrix**, 73  
    **polynom**, 65  
**unapply**, 12, 13, 85, 122, 162, 167,  
172, 188, 209, 215  
**unassign**, 8, 136, 183  
**union**, 20, 125, 130, 131, 158, 163,  
170, 201, 220, 226  
**units**, 10  
**value**, 115, 117, 118, 127, 144, 189,  
190, 209, 215  
**vector**, 19, 20, 71, 73, 74, 76  
**verboseproc**, 10  
**whattype**, 9  
    **while**, 8  
    **with**, 6, 18, 20  
**worksheet**, 6  
**xor**, 8  
    **data structure**, 7  
**function**  
    **specialized**, 6  
    **standard library**, 6  
    **help system**, 5  
    **incorrect response**, 9  
**letter**  
    **lower/upper-case**, 8  
**manipulation**  
    **of large expression**, 68  
**Maple**  
    Application Center, 4  
    basic concepts, 5  
    basic features, 4  
    comments, 10  
    Community, 4  
    interface, 4  
    kernel, 4  
    language, 7  
    library, 4  
    memory, 8  
    object, 7, 18  
        array, 19  
        list, 19  
        matrix, 19  
        sequence, 19  
        set, 19  
        table, 19  
        tensor, 85  
        vector, 19  
    palettes, 6  
    source code, 10, 18, 20  
    worksheet, 4, 6

Maplet  
    application, 6  
    user interface, 6  
module, 189

package  
    **DDEtools**, 178, 180, 181, 183, 185  
    **Discrete Transforms**, 164  
    **DiscreteTransforms**, 167  
    **LinearAlgebra**, 71–85, 127, 165,  
        166, 215, 220, 222  
    **PDEtools**, 163, 170, 187–191  
    **ScientificConstants**, 10  
    **Statistics**, 59, 91, 93  
    **Student[Calculus1]**, 109  
    **Student**, 109  
        Calculus1, 99  
    **Units**, 10  
    **Vector**  
        Calculus, 71–85, 127–134  
**diffforms**, 129

geom3d, 107  
geometry, 107  
inttrans, 155–170, 206–229  
linalg, 6, 18, 71–85, 127–134  
numapprox, 145  
orthopoly, 149  
plots, 20, 52, 64, 91–108, 113,  
    142, 165, 181, 183, 185, 188,  
    190  
plottools, 107  
powerseries, 120  
stats, 56, 59  
student, 109, 183  
tensor, 85–90  
subpackage, 6

stop button, 9

typesetting  
    **Standard Math**, 6

worksheet, 8

# *Mathematica* Index

- ' , 110, 154
- ( ), 31
- (\* \*), 31
- \*, 23, 27
- \*=, 27
- ++ , 27, 38, 39
- + =, 27
- , 27, 38
- =, 27
- >, 28, 37, 69, 77, 86, 114, 123, 150, 157, 158, 164, 172, 180
- . , 42, 43, 67, 74, 77, 79, 81–85, 88, 100, 128, 133, 222
- 
. , 28, 34, 69, 86, 92, 111–114, 123, 150, 157, 164, 207- // , 26, 31, 67, 69, 71, 74
- /., 29
- / ; , 29, 35, 57, 207, 221, 222
- /=, 27
- /@ , 26, 40, 44, 69, 82, 85
- :=, 28
- :>, 28, 164
- ; , 27, 48
- <, 27
- <<, 25, 44, 76
- <=, 27
- =, 28, 30
- =., 29
- ==, 27, 30, 38, 77
- == =, 30, 44, 71, 79, 81–84, 88, 138, 139, 153, 157
- >, 27
- >=, 27
- ? , 23, 29
- ?? , 23
- @ , 34, 35
- @@ , 44, 69, 221, 222
- [ ], 31
- [[ ]], 30, 37, 40, 44, 77, 86, 112, 137, 172, 178, 188
- #&, 32, 35, 37, 41, 44, 83, 84, 86, 123, 151, 157, 158, 160, 164
- % , 31
- &&, 27, 39, 67, 122, 123, 142, 143, 157, 161, 172, 227
- ^ :=, 28, 29
- ^=, 28, 29
- \_ , 28, 34, 35, 114
- \_?, 41, 77
- \_\_\_ , 86, 100
- { }, 31, 39, 173, 186
- ‘ ‘ , 31
- Abs , 28, 34, 35, 38, 122, 136, 142, 143, 162, 168
- AddTo , 27
- AffineShape , 108
- And , 27
- Animate , 61, 64, 190
- Apart , 69
- AppendColumns , 75
- AppendRows , 75, 82
- Append , 41, 63, 64, 81, 167
- Apply , 44, 69, 84, 86
- Arg , 136
- Array , 40–44, 72, 73, 78, 79, 81, 83, 86, 209, 216, 221, 222

Assuming, 154, 157, 159, 161, 162,  
214, 227  
BarChart3D, 59  
  AmbientLight, 60  
  AxesLabel, 60  
  BarLabels, 60  
  XSpacing, 60  
  YSpacing, 60  
BarChart, 59  
  BarEdges, 60  
  BarOrientation, 60  
  BarStyle, 60  
BesselI, 151  
BesselJZerosInterval, 151  
BesselJZeros, 151  
BesselJ, 151, 169  
BesselK, 151  
BesselYZeros, 151  
  WorkingPrecision, 151  
BesselY, 151, 162  
Beta, 148  
BlockMatrix, 78, 85  
Block, 36  
Break, 38  
Cancel, 69, 221, 222  
  Trig, 70  
CartesianMap  
  Lines, 139, 144, 151  
Cases, 41  
Catalan, 32  
Ceiling, 34  
CharacterRange, 40, 41  
Characteristic  
  Polynomial, 84  
Characters, 40  
ChebyshevT, 150  
ChebyshevU, 150  
Chop, 166, 167, 191  
Circle, 108  
ClearAll, 29  
ClearAttributes  
  Orderless, 221, 222  
Clear, 29  
CoefficientList, 66, 67, 84  
Coefficient, 66, 67, 125, 176, 210  
Collect, 66, 69, 125, 164, 176  
ColumnForm, 42, 72, 77  
Combine  
  trig, 34  
Compile, 26  
ComplexExpand, 136–138, 167  
  TargetFunctions, 136  
ComplexRootIntervals, 172  
Complex, 137  
Composition, 34  
Cone, 108  
Conjugate, 33, 136, 146  
Continue, 38  
ContourPlot3D, 103  
  Contours, 101  
ContourPlot, 103, 106  
  ContourLines, 103  
  ContourShading, 103, 104  
  Contours, 103, 104  
  PlotPoints, 104  
CosIntegral, 148  
CountRoots, 119, 172  
Cross, 74, 83, 100  
Cuboid, 108  
Curl, 128, 133  
Cylinder, 108  
CylindricalPlot3D, 106  
DSolve, 153, 157, 158, 160, 178–  
180, 182, 183, 188  
  GeneratedParameters, 178  
DTFourierTransform, 168  
Decrement, 27  
Degree, 32  
DeleteCases, 41, 77, 114  
Delete, 41  
Denominator, 69  
DensityPlot, 58  
Depth, 42  
Derivative, 110, 128–130

Det, 79, 81, 84, 86, 89, 216  
DiagonalMatrix, 78, 84  
DigitQ, 30  
Dimensions, 40, 42, 43, 77, 81, 86  
DiracDelta, 154, 157, 169  
Disk, 64, 134  
DisplayFunction, 48  
Distribute, 221  
DivideBy, 27  
Divide, 27  
Div, 128, 133, 134  
DotProduct, 128  
Dot, 74  
Do, 37–39, 44, 61, 63, 64, 84, 97, 98,  
  113, 114, 182, 185  
Drop, 41  
Dt, 110, 114, 128, 129  
  Constants, 129  
D, 29, 37, 89, 100, 148, 150, 154,  
  164, 188, 190, 207  
Eigensystem, 84  
Eigenvalues, 84  
Eigenvectors, 84  
Element, 69, 122, 123, 154, 172  
Eliminate, 131, 143  
EllipticE, 149  
EllipticF, 149  
EllipticK, 149  
EllipticPi, 149  
EngineeringForm, 26  
Equal, 27  
Erfc, 149  
Erf, 149, 157  
EulerGamma, 23, 32  
Evaluate, 58, 63, 161, 162, 186  
ExpIntegralEi, 162  
ExpIntegralE, 148  
ExpToTrig, 70, 210, 212  
ExpandAll, 69, 113, 164  
ExpandDenominator, 69  
ExpandNumerator, 69  
Expand, 23, 28, 69  
Exponent, 66, 67  
Exp, 33  
E, 32  
FactorTerms, 69  
Factor, 37, 69, 84, 117, 139  
  Extension, 69  
  GaussianIntegers, 69  
False, 30  
Fibonacci, 34  
Fib, 35  
File->Palettes, 24  
FilledListPlot, 91, 93  
  Fills, 76  
FilledPlot, 55, 119  
  Fills, 55  
FindMinimum, 110, 115  
FindRoot, 119, 173, 175  
  DampingFactor, 173, 174  
  Jacobian, 173  
  MaxIterations, 173, 174  
  WorkingPrecision, 173, 174  
First, 40, 77  
Flatten, 42, 84, 114, 123, 143  
Floor, 34  
Format->BackgroundColor, 24  
Format->Style, 24  
Format, 86  
For, 37, 38, 125, 167, 176  
FourierCosTransform, 162, 227  
FourierSinTransform, 162  
FourierTransform, 161, 164  
  FourierParameters, 161, 162  
FourierTrigSeries, 124  
Fourier, 164, 166, 167  
FractionalPart, 34  
FreeQ, 207, 221, 222  
FresnelC, 149  
FresnelS, 149  
FullForm, 26  
FullSimplify, 69, 169, 188, 207  
FunctionExpand, 69, 148, 152–154  
Function, 34

- GCD, 34  
Gamma, 148  
GegenbauerC, 150  
GeneralizedBarChart, 60  
GoldenRatio, 32  
Goto, 38  
Grad, 128, 133  
GramSchmidt, 83, 100  
    Normalized, 83  
Graphics3D, 76, 93, 108  
    Text, 53  
GraphicsArray, 60, 157, 166, 167  
Graphics, 58, 64, 108, 134  
    Text, 53  
GreaterEqual, 27  
Greater, 27  
HankelMatrix, 78  
HankelTransform, 169, 170  
Head, 26  
Help->WhyTheBeep, 23  
HelpBrowser, 23  
HermiteH, 150  
HilbertMatrix, 78  
Hold, 69  
HypergeometricOF1, 154  
Hypergeometric1F1, 154  
Hypergeometric2F1, 154  
HypergeometricPFQ, 154  
HypergeometricU, 154  
IdentityMatrix, 78, 79, 82, 84  
If, 36–38, 88, 114, 142, 210  
ImplicitPlot, 96, 175  
Implies, 27  
Im, 136, 138, 167  
Increment, 27  
Infinity, 32, 35, 227  
Inner, 74, 83, 87  
InputForm, 22, 26  
Input, 73  
Insert, 41, 82  
IntegerPart, 34  
IntegerQ, 29, 30, 34  
Integer, 75  
Integrate, 69, 83, 98, 139, 144,  
    148, 154, 207  
    Assumptions, 117  
    PrincipalValue, 117  
InterpolateRoot, 173  
Intersection, 42  
InverseDT  
    FourierTransform, 168  
InverseFourier, 164, 166, 167  
    Transform, 161, 162  
InverseLaplace  
    Transform, 155, 157, 159, 160,  
        225, 230  
InverseSeries, 70, 121  
Inverse, 79, 81, 84–86, 89  
I, 26, 32, 136  
JacobiP, 150  
Join, 38, 41, 142  
JordanDecomposition, 85  
LCM, 34  
LUBackSubstitution, 82  
LUDecomposition, 82  
LUMatrices, 82  
Label, 38  
LaguerreL, 150  
Laplace  
    Transform, 155, 157, 159, 160,  
        225, 230  
Laplacian, 128, 133  
Last, 40, 82  
LegendreP, 33, 150, 153  
LegendreQ, 153  
Length, 30, 37, 40, 42, 43, 77, 79,  
    81, 84, 86, 88, 89, 92, 114, 130,  
    137, 186, 221, 222  
LessEqual, 27  
Less, 27  
Level, 42  
Limit, 69, 110, 122, 139, 141, 146  
    Assumptions, 158, 162  
    Direction, 110

LinearEquations  
  ToMatrices, 81, 222

LinearSolve, 81

Line, 108

ListContourPlot3D, 103  
  Contours, 103  
  MeshRange, 103

ListContourPlot, 103  
  MeshRange, 103

ListDensityPlot, 58, 143  
  ColorFunction, 142  
  Mesh, 142

ListPlot3D, 91, 93  
  BoxRatios, 76  
  ColorFunction, 191  
  ViewPoint, 191

ListPlot, 44, 152, 168, 191  
  Frame, 76  
  PlotJoined, 91, 166, 167  
  PlotRange, 76, 122  
  PlotStyle, 76, 121, 122

ListPlotPlotStyle, 92

List, 39

LogIntegral, 148

LogLinearListPlot, 56

LogLinearPlot, 56

LogListPlot, 56, 57

LogLogListPlot, 56

LogLogPlot, 56, 57

LogPlot, 56

LogicalExpand, 27, 30

Log, 28, 33

LowerDiagonalMatrix, 78, 85

Map, 37, 44, 69, 73, 78, 79, 81, 82,  
  88, 89, 153, 157, 158, 160, 164,  
  230

MasterIndex, 23, 147  
  LoadingPackages, 25

MathematicalConstants, 32

MathematicalFunctions, 32

Operators, 27

Packages, 25

SpecialFunctions, 32

StandardPackages, 25

MathieuCPrime, 152

MathieuCharacteristicA, 52, 152

MathieuCharacteristicB, 52, 152

MathieuCharacteristic  
  Exponent, 152

MathieuC, 152

MathieuS, 152

MatrixExp, 74

MatrixForm, 43, 72, 73, 77–79, 81,  
  82, 84–86, 88, 89, 216

MatrixPlot  
  ColorFunction, 76

MatrixPower, 74, 84

MatrixQ, 73

MatrixRank, 81, 82

Minors, 79, 84

Minus, 27

Module, 36–38, 84, 86, 88, 89, 100,  
  113, 123, 143, 169, 227

Mod, 33, 34, 38

MovieParametricPlot, 62

MoviePlot, 61

MultipleListPlot, 51  
  PlotJoined, 52  
  SymbolShape, 63  
  SymbolStyle, 63

NDSolve, 179, 181, 187, 190  
  MaxSteps, 185

NDTFourierTransform, 168

NFourierTransform, 168

NIntegrate, 115, 128  
  WorkingPrecision, 117

NIInverseDTFourier  
  Transform, 168

NIInverseFourier  
  Transform, 168

NRoots, 66, 67, 173

NSolve, 55, 92, 173

Names, 25

NestList, 34, 35

- Nest, 34, 35, 118  
NormalDistribution, 57  
Normalize, 83  
  InnerProduct, 83  
Normal, 70, 121, 123, 124, 148  
Norm, 83  
Not, 27  
NullSpace, 77, 81, 84, 100  
NumberForm, 26, 58, 191  
Numerator, 69, 113  
N, 23, 26, 34, 38, 39, 42, 43, 58, 98,  
  126, 137, 142, 143, 154  
Off, 29  
Options[ContourPlot], 103  
Options[FindRoot], 173  
Options[Plot3D], 48  
Options[Plot], 48  
Options, 23  
Or, 27  
Outer, 74, 87  
PaddedForm, 43, 130, 131, 190, 191  
ParametricPlot3D, 94, 120  
  BoxRatios, 94  
  PlotPoints, 94  
ParametricPlot, 94, 97, 149, 160,  
  180, 183, 186  
  AspectRatio, 97  
Partition, 42, 142  
Part, 30, 40, 172  
PercentileBarChart  
  BarLabels, 60  
  BarSpacing, 60  
  BarStyle, 60  
PercentiledBarChart, 59  
PieChart, 59  
  PieExploded, 60  
  PieLabels, 60  
  PieStyle, 60  
PiecewiseExpand, 154, 158, 214  
Piecewise, 35, 57, 139, 157, 162,  
  214, 222  
Pi, 26, 32  
Plot3D, 22, 23, 47, 52, 61, 64, 190  
  Axes, 49  
  BoxRatios, 49  
  Boxed, 49  
  FaceGrids, 49  
  HiddenSurface, 49  
  Mesh, 49, 95  
  PlotPoints, 49, 104, 129  
  Shading, 49  
  TextStyle, 53  
  ViewPoint, 49  
PlotSymbol, 63  
PlotVectorField3D, 105  
PlotVectorField, 105, 181–183  
  AspectRatio, 106  
  Axes, 106  
  HeadLength, 106, 186  
  PlotStyle, 106  
Plot, 23, 35, 38, 44, 47, 61, 62, 64  
  AspectRatio, 49  
  AxesLabel, 49  
  AxesOrigin, 49  
  Axes, 49  
  DisplayFunction, 52  
  Evaluate, 52  
  FrameTicks, 49  
  Frame, 49  
  GridLines, 49, 54  
  PlotLabel, 49  
  PlotLegend, 49  
  PlotRange, 49, 99  
  PlotStyle, 49, 99  
  TextStyle, 53  
  Ticks, 49  
PlotDisplayFunction, 92  
PlotPlotRange, 92  
PlotPlotStyle, 92  
Plus, 27, 44, 74, 221, 222  
Point, 91, 93  
PolarDecomposition, 82  
PolarMap  
  Lines, 144

PolarPlot, 97  
  FrameTicks, 98  
  Frame, 98  
  PlotStyle, 98, 134  
PolyGamma, 148  
PolyLog, 148  
PolynomialGCD, 66, 84  
PolynomialQuotient, 66  
PolynomialQ, 34, 66  
PolynomialRemainder, 66  
PowerExpand, 69  
Power, 27, 28  
PreDecrement, 27  
PreIncrement, 27  
Prepend, 41  
Prime, 34, 97  
Print, 34, 38, 39, 84, 125, 138, 191,  
  226  
ProductLog, 150  
Product, 172  
Projection, 83  
  InnerProduct, 83  
Protect, 29  
PseudoInverse, 81  
Quotient, 34  
RGBColor, 76  
Random, 34, 41, 57, 58, 60, 74–78,  
  93, 103  
Range, 40, 67  
RealRootIntervals, 119, 172  
Real, 74, 76  
Reduce, 122, 150, 172  
Refine, 69  
Remove, 29  
ReplaceAll, 29  
ReplacePart, 41  
ReplaceRepeated, 29  
Residue, 145  
Rest, 41  
Return, 36  
Reverse, 41  
Re, 136, 138, 152, 166, 167  
Roots, 66, 67, 172  
Root, 32, 172  
RotateLeft, 41  
RotateRight, 41  
RotateShape, 108  
Round, 34  
RowReduce, 77, 81  
  Modulus, 82  
SameQ, 30  
ScatterPlot3D  
  Boxed, 76  
  PlotJoined, 63, 91, 93  
  PlotRange, 76  
  PlotStyle, 76, 91, 93  
ScientificForm, 26, 32  
Select, 41, 123  
SequenceForm, 86  
Sequence, 86  
SeriesCoefficient, 121  
Series, 70, 124, 145, 148, 150, 154  
Set2DJulia, 142  
Set3DJulia, 142  
Set3DMandelbrot, 143  
SetCoordinates, 128, 133, 134  
SetDelayed, 28  
SetOptions, 50, 62, 151, 152  
Set, 28  
Short, 69  
ShowAnimation, 44, 61–64  
Show, 52, 53, 57, 58, 64, 76, 91–93,  
  108, 134, 167, 179, 182, 183,  
  185, 186  
  AspectRatio, 130  
  Background, 52  
  ViewPoint, 95, 130  
Signature, 86, 88  
Sign, 34, 136  
Simplify, 34, 67, 69, 81  
SinIntegral, 148  
Solve, 23, 29, 77, 81, 84, 114, 130,  
  132, 145, 157, 160, 207  
VerifySolutions, 172

Sort, 41  
SparseArray, 77, 78, 86, 89  
Sphere, 108  
SphericalPlot3D, 107  
Sqrt, 26  
StackedBarChart, 59  
    BarStyle, 60  
StringDrop, 31  
StringForm, 31  
StringInsert, 31  
StringJoin, 31  
StringLength, 31  
StringPosition, 31  
StringReplace, 31  
StringReverse, 31  
StringTake, 31  
StyleForm, 58  
SubMatrix, 75, 82  
Subscript, 86  
Subset, 42  
SubtractFrom, 27  
Subtract, 27  
Sum, 36–38, 44, 116, 118, 121, 122,  
    126, 141, 154, 167, 216  
Superscript, 86  
SurfaceOfRevolution  
    RevolutionAxis, 104  
Switch, 37, 38  
TableForm, 43, 74, 130, 131, 179,  
    190, 191  
    TableHeadings, 124  
Table, 26, 35, 40, 41, 43, 44, 52,  
    57, 58, 60, 62, 63, 67, 72–79,  
    83–86, 88, 89, 103, 119, 121,  
    122, 124, 125, 130, 137, 138  
TakeColumns, 75  
TakeMatrix, 75  
TakeRows, 75, 81  
Take, 41, 63, 72, 73  
TensorQ, 86, 89  
TensorRank, 40, 42, 43, 86  
Text, 58  
Thread, 44, 69, 77, 81  
TimesBy, 27  
Times, 27, 74, 221, 222  
Timing, 26, 34, 35, 168  
Together, 69  
TraditionalForm, 154  
TranslateShape, 108  
Transpose, 77–79, 81, 82, 84, 87,  
    100, 191  
TreeForm, 26, 42, 43, 86  
Triangle, 63  
Triangular  
    Matrix, 78  
TrigExpand, 70, 71  
TrigFactor, 70  
TrigReduce, 28, 70, 125  
TrigToExp, 70  
True, 29, 30  
Tr, 84  
Unequal, 27  
Union, 42, 77, 88, 125, 130, 164,  
    221, 222, 226  
UnitStep, 35, 127, 154, 159, 160  
Unprotect, 29  
UpperDiagonalMatrix, 85  
UpperDiagonal  
    Matrix, 78  
Variables, 66  
VectorField, 128  
VectorQ, 42, 72  
Which, 37, 38  
While, 37–39, 142, 143  
WireFrame, 108  
With, 36  
WorldPlot, 25  
Xor, 27  
ZeroMatrix, 78, 85  
Zeta, 154  
\$ContextPath, 25  
\$Context, 25  
\$Packages, 25  
\$RecursionLimit, 35

\_Complex, 26  
\_Integer, 26  
\_Real, 26  
adjoint, 79  
angle, 83  
augmentedMat, 81, 82  
blockJordan, 85  
characteristicMat, 84  
cofactor, 79  
columnSpace, 77  
commutatorRelation, 88  
context'short, 25, 31  
corank, 81  
doubleFactorial, 38  
ecLagrange, 37  
eliminationGaussian, 82  
generateEqs, 81  
generateMat, 81  
help, 23  
linSec, 99  
linTan, 99, 113  
linearCombination, 87  
linearDependence, 77  
mF, 36  
mIJ, 79  
minimalPolynomial, 84  
pMaclaurin, 123  
productKronecker, 87  
productLie, 88  
rank, 81  
regPertPoly, 176  
rowReduceMat, 81, 82  
rowSpace, 77  
sChristophel1, 89  
sChristophel2, 89  
solGeneral, 81  
solParticular, 81  
tLeviCivita, 88  
tensorForm, 86  
theoremCayleyHamilton, 84  
trHankelH1, 151  
trHankelH2, 151

tubePlot1, 100  
tubePlot2, 100  
vConj, 139

comments, 31  
contexts, 25, 31  
current, 25  
list of, 25  
sequence of, 25

data structure, 26

help system, 23

Initialization, 30

Mathematica  
basic concepts, 23  
basic features, 22  
basic principles, 26  
cells, 24  
design, 22  
front end, 22, 24  
history, 22  
interface, 22  
kernel, 22  
language, 26  
manual, 23  
notebook, 22, 24  
objects, 26, 39  
palettes, 24, 43

mathematical  
constants, 32  
typesetting system, 22

MathLink protocol, 22

menu  
Cell, 61  
File, 24  
Format, 24  
Help, 23  
Input, 43, 48, 50  
Kernel, 32

- package, 25
  - Algebra**
    - RootIsolation**, 119, 172
  - DiscreteMath**
    - Combinatorica**, 42
  - Graphics3D**
    - Shapes**, 25
  - Graphics**
    - Animation**, 44, 61–64, 190
    - Colors**, 48
    - ComplexMap**, 144, 151
    - ContourPlot3D**, 101, 103
    - FilledPlot**, 91–108
    - Graphics3D**, 91–108
    - Graphics**, 56, 91–108, 134
    - ImplicitPlot**, 96, 175
    - Legend**, 49
    - MultipleListPlot**, 63
    - ParametricPlot3D**, 106, 107
    - PlotField**, 105, 180, 182, 183, 186
    - Shapes**, 108
  - IneqGraphics**, 56
  - IneqPlot**, 56
  - LinearAlgebra**
    - MatrixManipulation**, 72–89
    - Orthogonalization**, 83, 100
- Miscellaneous**
  - ChemicalElements**, 32
  - PhysicalConstants**, 32
  - Units**, 32
  - WorldPlot**, 25
- NumericalMath**
  - BesselZeros**, 151
  - InterpolateRoot**, 173
- Statistics**, 57
- VectorAnalysis**, 128–134
- list of functions in, 25
- loaded, 25

RGB value, 48

short names, 25

Symbolic

Manipulation Program, 22

transformation rules, 28

Wolfram

Information Center, 56

Research, 22

Wolfram, Stephen, 22

## *Springer und Umwelt*

ALS INTERNATIONALER WISSENSCHAFTLICHER VERLAG sind wir uns unserer besonderen Verpflichtung der Umwelt gegenüber bewusst und beziehen umweltorientierte Grundsätze in Unternehmensentscheidungen mit ein.

VON UNSEREN GESCHÄFTSPARTNERN (DRUCKEREIEN, Papierfabriken, Verpackungsherstellern usw.) verlangen wir, dass sie sowohl beim Herstellungsprozess selbst als auch beim Einsatz der zur Verwendung kommenden Materialien ökologische Gesichtspunkte berücksichtigen.

DAS FÜR DIESES BUCH VERWENDETE PAPIER IST AUS chlorfrei hergestelltem Zellstoff gefertigt und im pH-Wert neutral.